

[illegible]

```
CCCCCCCC NN NN XX XX MM MM AAAAAA NN NN
CCCCCCCC NN NN XX XX MM MM AAAAAA NN NN
CC NN NN XX XX MMMM MMMM AA AA NN NN
CC NN NN XX XX MM MM AA AA NN NN
CC NNNN NN XX XX MM MM AA AA NNNN NN
CC NNNN NN XX XX MM MM AA AA NNNN NN
CC NN NN XX XX MM MM AA AA NN NN
CC NN NN XX XX MM MM AA AA NN NN
CC NN NNNN XX XX MM MM AAAAAAAAAA NN NNNN
CC NN NNNN XX XX MM MM AAAAAAAAAA NN NNNN
CC NN NN XX XX MM MM AA AA NN NN
CC NN NN XX XX MM MM AA AA NN NN
CCCCCCCC NN NN XX XX MM MM AA AA NN NN
CCCCCCCC NN NN XX XX MM MM AA AA NN NN
```

```
....
....
....
....
```

```
LL IIIIII SSSSSSSS
LL IIIIII SSSSSSSS
LL II
LL II
LL II
LL II
LL II
LL II
LL II
LL II
LL II
LL II
LL II
LLLLLLLLLL IIIIII SSSSSSSS
LLLLLLLLLL IIIIII SSSSSSSS
```

(2)	175	DECLARATIONS
(4)	310	CNX\$INIT - Initialize connection manager
(5)	419	CNX\$NEWSYSTEM - New system heard from
(6)	474	CNX\$NEW_CSB - Connect to new system
(7)	512	CNX\$CONNECT - Connect to remote system
(8)	670	CNX\$RCV_CNCT_MSG - Receive CONNECT message
(9)	838	CNX\$ACCEPT - Accept connection from remote system
(10)	956	CNX\$DISC_BUGCHECK - Disconnect from Node and Request it to Bugcheck
(10)	957	CNX\$DISC_REMOVE - Disconnect from Node Removed from Cluster
(10)	958	CNX\$DISC_PROTOCOL - Disconnect from Node for protocol reasons
(11)	1029	CNX\$ERROR - Connection error
(12)	1082	CNX\$BREAK - Cleanup and Disconnect SCS Connection
(13)	1138	CNX\$DISCONNECT - Disconnect from remote system
(14)	1201	CNX\$WAIT - Initiate timeout
(15)	1300	CNCT_DATA - Setup Connect Data in CSB
(16)	1350	CNCT_CHECK - Verify Connect Data
(17)	1439	RECNT_CHECK - Verify Reconnect Data
(18)	1556	LONG_BREAK - Long Break in Connection
(19)	1594	CNX\$DECREFCNT - Decrement CSB Reference Count
(20)	1647	DELETE_TQE - Delete a TQE linked to a CSB
(21)	1690	DEAD_NODE - Manage death of a node
(22)	1731	CNX_STATUS_CHECK - Check SCS failure message
(23)	1795	CNX\$LOOKUP_CSB - Lookup a CSB given a SB address
(24)	1867	CNX\$CREATE_CSB - Create a new CSB given a SB address
(25)	1976	DISPATCH - Dispatch on CSB state


```
0000 1 .TITLE CNXMAN - Cluster Connection Manager
0000 2 .IDENT 'V04-000'
0000 3
0000 4 *****
0000 5
0000 6 *
0000 7 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 8 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 9 * ALL RIGHTS RESERVED.
0000 10 *
0000 11 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 12 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 13 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 14 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 15 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 16 * TRANSFERRED.
0000 17 *
0000 18 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 19 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 20 * CORPORATION.
0000 21 *
0000 22 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 23 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 24 *
0000 25 *****
0000 26
0000 27
0000 28 **
0000 29 FACILITY: EXECUTIVE, CLUSTER MANAGEMENT
0000 30
0000 31 ABSTRACT:
0000 32 This module creates and manages connections to the other systems
0000 33 in the cluster.
0000 34
0000 35 ENVIRONMENT: VAX/VMS
0000 36
0000 37 AUTHOR: Steve Beckhardt, CREATION DATE: 17-Aug-1982
0000 38
0000 39 MODIFIED BY:
0000 40
0000 41 V03-028 DWT0235 David W. Thiel 7-Aug-1984
0000 42 Correct source of SCS message size for testing.
0000 43
0000 44 V03-027 DWT0228 David W. Thiel 24-Jul-1984
0000 45 Modify send credits from 10 to 5. Modify limit
0000 46 on unacknowledged messages from 7 to 4.
0000 47
0000 48 V03-026 DWT0216 David W. Thiel 30-Apr-1984
0000 49 Correct sequencing of events for "lasp gasp"
0000 50 messages.
0000 51
0000 52 V03-025 DWT0206 David W. Thiel 07-Apr-1984
0000 53 Add support for "Last Gasp" from a failing system.
0000 54 Initialize CLUBPW block in CLUB.
0000 55
0000 56 V03-024 DWT0202 David W. Thiel 25-Mar-1984
0000 57 Remove all references to CNCT$V_QOURUM and
```

```
0000 58 : CNCT$V_TRANSITION.
0000 59 :
0000 60 : V03-023 DWT0191 David W. Thiel 21-Mar-1984
0000 61 : Update to support new ACKMSG. Reinstate improved
0000 62 : version number checking.
0000 63 :
0000 64 : V03-022 DWT0176 David W. Thiel 23-Feb-1984
0000 65 : Initialize CLUB$W_QDVOTES to largest integer when
0000 66 : creating CLUB. Maintain SB$CL_CSB as a pointer to
0000 67 : the newest CSB for a system.
0000 68 :
0000 69 : V03-021 DWT0163 David W. Thiel 19-Jan-1984
0000 70 : Correct CNX$DISC_* routines. Support forced
0000 71 : disconnection in the general case. Rename
0000 72 : CNX_ERROR to CNX$ERROR.
0000 73 :
0000 74 : V03-020 DWT0148 David W. Thiel 13-Dec-1983
0000 75 : Store SYSGEN parameters LCKDIRWT and QDSKVOTES in
0000 76 : the local CSB. Restructure code. Use CNX$ALLOZMEM
0000 77 : to allocate and zero pool. Correct disabling of
0000 78 : polling once a node is firmly discovered.
0000 79 :
0000 80 : V03-019 DWT0142 David W. Thiel 07-Nov-1983
0000 81 : Use symbolic protocol level (CNCT$K_PROTOCOL).
0000 82 :
0000 83 : V03-018 DWT0127 David W. Thiel 30-Aug-1983
0000 84 : Pull console message routines out into new module
0000 85 : CLUMESSAG.MAR.
0000 86 : Disable process polling after accepting a connection.
0000 87 : Check more carefully for a fatal disconnect.
0000 88 :
0000 89 : V03-017 DWT0117 David W. Thiel 24-Aug-1983
0000 90 : Replace systemid with node name in all messages.
0000 91 : Change CONFIG_CHANGE to CNX$CONFIG_CHANGE. Update
0000 92 : protocol version level to 6 to mark incompatibility
0000 93 : with previous versions.
0000 94 :
0000 95 : V03-016 DWT0116 David W. Thiel 2-Aug-1983
0000 96 : Increment protocol level to mark incompatibility
0000 97 : with previous versions.
0000 98 :
0000 99 : V03-015 DWT0109 David W. Thiel 16-Jul-1983
0000 100 : Use CNX$CHECK_QUORUM to hang on lose of quorum.
0000 101 : Tolerate repeating software incarnation numbers.
0000 102 : Correct cleanup after an ACCEPT fails. Clean up
0000 103 : code a little bit. Improve some messages.
0000 104 :
0000 105 : V03-014 ROW0185 Ralph D. Weber 21-JUN-1983
0000 106 : Change CSB SEL queue to block transfer partners BTX queue, to
0000 107 : support connection manager block transfers. Remove CLUB
0000 108 : references to SEL queue.
0000 109 :
0000 110 : V03-013 DWT0105 David W. Thiel 16-Jun-1983
0000 111 : Fail message on any call to LONG_BREAK.
0000 112 : Refuse to open connection when LONG_BREAK is set.
0000 113 :
0000 114 : V03-012 DWT0098 David W. Thiel 14-May-1983
```


0000	115	:	
0000	116	:	If incompatible connection manager's see each other,
0000	117	:	BUGCHECK one of them.
0000	118	:	Remove temporary configuration management code and
0000	119	:	integrate use of CONMAN module.
0000	120	:	Move CNX\$DISPATCH to CONMAN.
0000	121	:	Add CSB\$L SB as pointer to SB.
0000	122	:	More initialization of local CSB.
0000	123	:	Dynamically allocate CLUB structure.
0000	124	:	
0000	125	:	V03-011 ROW0186 Ralph O. Weber 25-APR-1983
0000	126	:	Bump protocol version number to indicate use of two level
0000	127	:	dispatching. Add setup for CLUB\$L JNL DISPT in CNX\$INIT.
0000	128	:	Change LCK\$GL_DIRSYS\$CSB to LCK\$GL_DIRSYS\$CSID. Change setup to
0000	129	:	put directory node CSID in there. Add CNX\$DISPATCH, the
0000	130	:	target of the first level input dispatcher for FAC_CNX
0000	131	:	messages.
0000	132	:	
0000	133	:	V03-010 DWT0093 David W. Thiel 15-Apr-1983
0000	134	:	Track changes in \$CLUBDEF.
0000	135	:	
0000	136	:	V03-009 DWT0090 David W. Thiel 31-Mar-1983
0000	137	:	Add reconnection data to detect partitioned clusters.
0000	138	:	Extend CSB and CLUB.
0000	139	:	Change protocol version to 2.
0000	140	:	
0000	141	:	V03-008 DWT0085 David W. Thiel 14-Mar-1983
0000	142	:	Avoid attempt to output message during initialization.
0000	143	:	Correct misuse of stack for connect data.
0000	144	:	
0000	145	:	V03-007 DWT0084 David W. Thiel 12-Mar-1983
0000	146	:	Correct bug that allows a reconnect to be sent to
0000	147	:	a recently rebooted system. Log CSB creations.
0000	148	:	
0000	149	:	V03-006 SRB0070 Steve Beckhardt 10-Mar-1983
0000	150	:	Added routine to send the job controller a message
0000	151	:	when a system is removed from the cluster. This is
0000	152	:	a temporary change.
0000	153	:	
0000	154	:	V03-005 DWT0083 David W. Thiel 10-Mar-1983
0000	155	:	Replace HALTs with generic connection manager
0000	156	:	BUG CHECKS. Create and use CLUSTER Block. Change
0000	157	:	SYSAP name.
0000	158	:	
0000	159	:	V03-004 DWT0082 David W. Thiel 3-Mar-1983
0000	160	:	Correct use of disconnect/reject status codes.
0000	161	:	
0000	162	:	V03-003 DWT0070 David W. Thiel 21-Feb-1983
0000	163	:	Major revision which includes:
0000	164	:	Initialize automatically on being loaded.
0000	165	:	Use SCA Process Poller to find new systems.
0000	166	:	More state oriented structure.
0000	167	:	Split out acknowledged message services as ACKMSG.
0000	168	:	
0000	169	:	V03-002 SRB0064 Steve Beckhardt 21-Jan-1983
0000	170	:	Removed cell LCK\$GL_DIRSYS\$CSB as it now resides in the
0000	171	:	EXEC (in module SYSCOMMON).

CNXMAN
V04-000

- Cluster Connection Manager

I 8

16-SEP-1984 00:24:50 VAX/VMS Macro V04-00
5-SEP-1984 04:07:15 [SYSLOA.SRC]CNXMAN.MAR;1

Page 4
(1)

0000 172 :--
0000 173

```
0000 175      .SBTTL  DECLARATIONS
0000 176      :
0000 177      : INCLUDE FILES:
0000 178      :
0000 179      $CDTDEF      : CDT Offsets
0000 180      $CLMDRDEF    : Cluster disconnect/reject codes
0000 181      $CLMSGDEF    : Cluster message definitions
0000 182      $CLUBDEF     : CLuster Block offsets
0000 183      $CNCTDEF     : CONNECT data offsets
0000 184      $CSBDEF      : CSB Offsets
0000 185      $DYNDEF      : Data structure type codes
0000 186      $FKBDEF      : Fork block offsets
0000 187      $IPLDEF      : IPL definitions
0000 188      $MSGDEF       : Mailbox message type codes
0000 189      $PBDDEF      : PB Offsets
0000 190      $PDTDEF      : PDT Offsets
0000 191      $SBDEF        : SB Offsets
0000 192      $SCSCMGDEF   : SCS CONNECT Message definitions
0000 193      $TQEDEF      : TQE offsets
0000 194      :
0000 195      : MACROS:
0000 196      :
0000 197      :
0000 198      :
0000 199      .MACRO  STATE_DISP      LIST
0000 200      BSBW    DISPATCH
0000 201      .IRP    ITEM,<LIST>
0000 202      STATE_ENTRY  ITEM
0000 203      .ENDR
0000 204      .BYTE    0                : Flag end
0000 205      .ENDM    STATE_DISP
0000 206      :
0000 207      .MACRO  STATE_ENTRY      CODE,ADDR,?LABEL
0000 208      .BYTE    CSB$K-1 CODE
0000 209      .SIGNED_WORD  ADDR-LABEL
0000 210 LABEL: .ENDM  STATE_ENTRY
0000 211      :
0000 212      : EQUATED SYMBOLS:
0000 213      :
0000 214      :
0000 215      :
00000005 0000 216 SEND_CREDITS = 5                : Connection send credits
0000 217      :
0000 218      : The unacknowledged message limit is set to SEND_CREDITS-1
0000 219      : in order to force an explicit acknowledgement message to
0000 220      : be sent BEFORE SCS will ever consider sending an explicit
0000 221      : credit message. This is done because the acknowledgement
0000 222      : message costs the same as the credit message and does
0000 223      : more (it completes sending threads). This formula for
0000 224      : choosing an unacknowledged message limit is conservative
0000 225      : in that message buffers cached in the warm CDRP cache
0000 226      : instead of being deallocated cannot cause explicit SCS
0000 227      : credit messages.
0000 228      :
0000 229      :
0000 230      :
0000 231      : OWN STORAGE:
```



```
0000 232 ;
0000 233 ;
0000 234 .PSECT $$$040, LONG ; R/W Data PSECT
0000 235
0000 236 PROC_NAME: ; SYSAP name
74 73 75 6C 63 58 41 56 24 53 4D 56 0000 237 .ASCII 'VMS$VAXcluster'
72 65 000C 238
000E 239 .REPT 16, <.-PROC_NAME>
000E 240 .ASCII
20 000E 241 .ENDR
0010 242 *****
0010 243
0010 244 : NOTE: The following assumptions are in effect for this entire module.
0010 245 :
0010 246 : *****
0010 247
0010 248 ASSUME IPL$_SYNCH EQ IPL$_SCS
0010 249 ASSUME IPL$_SYNCH EQ IPL$_TIMER
0010 250
0010 251 .DEFAULT DISPLACEMENT, WORD
```

```
0010 253 :++
0010 254
0010 255 STATE-ORIENTED DESCRIPTION OF CONNECTION MANAGEMENT LAYER:
0010 256
0010 257
0010 258 The connection manager is organized as a state machine.
0010 259 Each connection has its own independent state machine. Each
0010 260 connection is represented by a Connection Status Block (CSB).
0010 261 The state of a connection is defined by the contents of the
0010 262 CSB$B STATE field of the CSB. The states are defined by
0010 263 symbols of the format: CSB$K_state where "state" is the
0010 264 state name.
0010 265
0010 266 STATES:
0010 267
0010 268 NEW
0010 269 Brand new connection block created as the result of a
0010 270 reference to a node id/software incarnation for which no
0010 271 CSB existed.
0010 272
0010 273 CONNECT
0010 274 Initial connect request to a newly discovered system in
0010 275 progress.
0010 276
0010 277 ACCEPT
0010 278 Initial connection from a newly discovered system being
0010 279 accepted.
0010 280
0010 281 OPEN
0010 282 Connection to a system exists and is available for use.
0010 283 This is the "normal" state of a CSB.
0010 284
0010 285 DISCONNECT
0010 286 Disconnect of an open connection in progress.
0010 287
0010 288 WAIT
0010 289 Timeout in progress. On conclusion of the timeout, an
0010 290 attempt will be made to reconnect to the remote system.
0010 291
0010 292 RECONNECT
0010 293 Connect in progress to a system to which a previous
0010 294 connection broke.
0010 295
0010 296 REACCEPT
0010 297 Accept in progress to a system to which a previous
0010 298 connection broke.
0010 299
0010 300 DEAD
0010 301 A new incarnation of the node has been seen.
0010 302 No connection new connection to the incarnation specified
0010 303 by the CSB is possible, for obvious reasons.
0010 304
0010 305 LOCAL
0010 306 Special state only found for the local node.
0010 307
0010 308 :--
```

```
0010 310 .SBTTL CNX$INIT - Initialize connection manager
0010 311
0010 312
0010 313
0010 314
0010 315
0010 316
0010 317
0010 318
0010 319
0010 320
0010 321
0010 322
0010 323
0010 324
0010 325
0010 326
0010 327
0010 328
0010 329
0010 330
0010 331
0010 332
0010 333
0010 334
0010 335
0010 336
0010 337
0010 338
0010 339
0010 340
0010 341
00000000 342 .PSECT $$$002 ; Initialization section
0000 343
0000 344 CNX$INIT::
51 01A8 8F 3C 0000 345 MOVZWL #CLUB$K_LENGTH,R1 ; Length of CLUB
FFF8' 30 0005 346 BSBW CNX$ALLOZMEM ; Allocate and zero memory
3D 50 E9 0008 347 BLBC R0,5$ ; Branch if memory not available
00000000'GF 52 D0 000B 348 MOVL R2,G*CLUB$GL CLUB ; Store for the world to see
62 62 7E 0012 349 MOVAQ CLUB$CL_CSBQFL(R2), - ; Initialize CSB queue
04 A2 62 7E 0015 350 MOVAQ CLUB$CL_CSBQFL(R2), -
0B A2 03 90 0019 351 MOVAQ CLUB$CL_CSBQBL(R2), -
001D 352 CLUB$CL_CSBQBL(R2)
0B A2 03 90 0019 353 MOVB #DYN$C-CLU CLUB, - ; Block subtype
001D 354 CLUB$B-SUBTYPE(R2)
14 A2 0000'CF DE 001D 355 MOVAL W*CLUB$MIN_JOURNAL,CLUB$CL_JNL_DISPT(R2) ; Init journal dispatch
66 A2 01 B0 0023 356 MOVW #1,CLUB$W-FIRST_INDEX(R2) ; Next available CSID slot
00AE C2 00 B2 0027 357 MCOMW #0,CLUB$W-QDVOTES(R2) ; Initialize to maximum possible votes
002C 358 ASSUME CLUB$FB$-FORK_BLOCK GE FKBSK_LENGTH
50 00CC C2 9E 002C 359 MOVAB CLUB$B-FORK_BLOCK(R2),R0 ; Address of fork block
0B A0 0B 90 0031 360 MOVB #IPL$ SCS,FKBSB FIPL(R0) ; Store IPL in fork block
0035 361 ASSUME CLUB$P$-FORK_BLOCK GE FKBSK_LENGTH
50 018C C2 9E 0035 362 MOVAB CLUB$B-CLUBPWFT(R2),R0 ; Power recovery fork block
0B A0 0B 90 003A 363 MOVB #IPL$ SCS,FKBSB FIPL(R0) ; Store IPL in fork block
51 00000000'GF DE 003E 364 MOVAL G*SCS$GA_LOCALSB,R1
0598' 30 0045 365 BSBW CNX$CREATE_CSB ; Create CSB for local system
63 50 E9 0048 366 5$: BLBC R0,7$ ; Branch on error
```


	51	64	A5	D0	004B	367	MOVL	CSBSL CLUB(R5),R1	:	Get address of cluster block	
	10	A1	55	D0	004F	368	MOVL	R5,CLOB\$ LOCAL CSB(R1)	:	Local system CSB address	
	50	7C	A5	9E	0053	369	MOVAB	CSBSB CNCT(R5),R0	:	Connect data template	
	40	A5	60	90	0057	370	MOVB	CNCTSB ECOLVL(R0),-	:	Fill in protocol ECO level	
					005B	371		CSBSB ECOLVL(R5)			
	41	A5	01	A0	90	005B	372	MOVB	CNCTSB VERNUM(R0),-	:	Fill in protocol version
					0060	373		CSBSB VERNUM(R5)			
	50	A5	00000000'GF	B0	0060	374	MOVW	G^CLUSGW VOTES,-	:	Votes held by local system	
					0068	375		CSBSW VOTES(R5)			
	52	A5	00000000'GF	B0	0068	376	MOVW	G^CLUSGW QUORUM,-	:	Local system proposed quorum	
					0070	377		CSBSW QUORUM(R5)			
	56	A5	00000000'GF	B0	0070	378	MOVW	G^CLUSGW QDSKVOTES,-	:	Local system proposed quorum disk votes	
					0078	379		CSBSW QDVOTES(R5)			
	54	A5	00000000'GF	B0	0078	380	MOVW	G^CLUSGW LCKDIRWT,-	:	Lock manager directory system weight	
					0080	381		CSBSW LCKDIRWT(R5)			
	60	A5	01000000 8F	C8	0080	382	BISL2	#CSBSM LOCAL,-	:	Mark this the local CSB	
					0088	383		CSBSL STATUS(R5)			
	43	A5	0B	90	0088	384	MOVB	#CSBSK LOCAL,-	:	Set state of local CSB	
					008C	385		CSBSB_STATE(R5)			
					008C	386					
					008C	387		: LISTEN for incoming CONNECTS.			
					008C	388					
					008C	389					
					008C	390	LISTEN	MSGADR = CNX\$RCV CNCT_MSG,-	:	Listen for incoming CONNECTs	
					008C	391		LPRNAM = PROC_NAME,-			
					008C	392		PRINFO = (R0),-	:	Use data in local CSB	
					008C	393		ERRADR = LISTEN_ERROR			
		49	50	E9	00A3	394	BLBC	R0,10\$:	Branch on failure	
	51	040C	8F	3C	0GA6	395	MOVZWL	#12+<256*4>,R1	:	Length of cluster vector	
		FF52'		30	00AH	396	BSBW	CNX\$ALLOZMEM	:	Allocate and zero memory	
		3E	50	E9	00AE	397	BLBC	R0,10\$:	Branch on failure	
		OB	A2	02	90	00B1	MOVW	#DYN\$C CLU CLUEVC,11(R2)	:	Store sub-type	
	00000000'GF	OC	A2	9E	00B5	398	MOVAB	12(R2)-G^CLUSGL CLUSVEC	:	Store vector address	
	00000000'GF	0100	8F	B0	00BD	399	MOVW	#256,G^CLUSGW_MAXINDEX	:	Maximum cluster vector index+1	
					00C6	400					
	50	0007'CF		9E	00C6	401	MOVAB	W^CNX\$NEWSYSTEM,R0	:	Address of new system routine	
	52	0000'CF		9E	00CB	402	MOVAB	PROC_NAME,R2	:	Address of process name	
		00000000'GF		16	00D0	403	JSB	G^SCS\$POLL_PROC	:	Poll for copies of self	
		16	50	E9	00D6	404	BLBC	R0,10\$:	Branch on failure	
	55	00000000'GF		D0	00D9	405	MOVL	G^CLUSGL CLUB,R5	:	Get address of cluster block	
		OC	A5	51	D0	00E0	MOVL	R1,CLUB\$C_POLL_CTX(R5)	:	Save context for later calls	
					00E4	407			:	R0 is odd -- enable polling	
					00E4	408			:	R1 is address of SPPB	
		52		D4	00E4	409	CLRL	R2	:	All systems now and forever	
	00000000'GF			16	00E6	410	JSB	G^SCS\$POLL_MODE	:	Enable polling	
		FF11'		30	00EC	411	BSBW	CNX\$CON_INIT	:	Initialize configuration manager	
				05	00EF	412	RSB		:	Return status	
					00F0	413					
					0000	0000	.PSECT	\$\$\$100,LONG			
					0000	415	LISTEN_ERROR:				
					0000	416	DISCONNECT		:	Clean up error	
					05	0006	RSB		:	and return	

```
0007 419 .SBTTL CNX$NEWSYSTEM - New system heard from
0007 420
0007 421 :++
0007 422 :
0007 423 FUNCTIONAL DESCRIPTION:
0007 424 :
0007 425 This routine is called when the SCA Process Poller detects
0007 426 a possible new system.
0007 427 :
0007 428 CALLING SEQUENCE:
0007 429 :
0007 430 JSB CNX$NEWSYSTEM
0007 431 IPL must be at IPL$_SYNCH
0007 432 :
0007 433 INPUT PARAMETERS:
0007 434 :
0007 435 R0 is context (nothing interesting)
0007 436 R1 is address of process name
0007 437 R2 is address of system ID
0007 438 R3 is address of process information
0007 439 :
0007 440 OUTPUT PARAMETERS:
0007 441 :
0007 442 NONE
0007 443 :
0007 444 COMPLETION CODES:
0007 445 :
0007 446 R0 odd = success
0007 447 Disable polling on the system
0007 448 R0 even = failure
0007 449 Continue polling the system
0007 450 :
0007 451 SIDE EFFECTS:
0007 452 :
0007 453 R1-R5 are destroyed
0007 454 :
0007 455 :--
0007 456 :
00000007 457 .PSECT $$$100, LONG
0007 458 :
0007 459 CNX$NEWSYSTEM::
51 52 D0 0007 460 MOVL R2,R1 ; Address of System ID
52 D4 000A 461 CLRL R2 ; No buffer
00000000'GF 16 000C 462 JSB G^SCSS$CONFIG_SYS ; Find SB
19 50 E9 0012 463 BLBC R0,30$ ; Branch on not found
057E 30 0015 464 BSBW CNX$LOOKUP_CSB ; Find or allocate a CSB
13 50 E9 0018 465 BLBC R0,30$ ; Branch on invalid system ID
001B 466 STATE_DISP <<NEW,20$>,<CONNECT,30$>,<ACCEPT,30$>>
50 01 D0 0028 467 10$: MOVL #1,R0 ; Disable polling
05 002B 468 RSB ; Return
002C 469 :
03 10 002C 470 20$: BSBB CNX$NEW_CSB ; Do initial connect to new CSB
50 D4 002E 471 30$: CLRL R0 ; Continue polling
05 0030 472 RSB ; Unable to allocate memory
```

```
0031 474 .SBTTL CNX$NEW_CSB - Connect to new system
0031 475
0031 476 :++
0031 477
0031 478 FUNCTIONAL DESCRIPTION:
0031 479
0031 480 This routine is called to do the initial connect to a
0031 481 new system.
0031 482
0031 483 CALLING SEQUENCE:
0031 484
0031 485 JSB CNX$NEW_CSB
0031 486 IPL must be at IPL$SYNCH
0031 487
0031 488 INPUT PARAMETERS:
0031 489
0031 490 R5 is the CSB address
0031 491
0031 492 OUTPUT PARAMETERS:
0031 493
0031 494 NONE
0031 495
0031 496 COMPLETION CODES:
0031 497
0031 498 NONE
0031 499
0031 500 SIDE EFFECTS:
0031 501
0031 502 R0-R4 are destroyed
0031 503
0031 504 :--
0031 505
0031 506 CNX$NEW_CSB::
43 05 90 0031 507 MOVB #CSB$K_CONNECT, - ; Set state
AS 0031 508 CSB$B_STATE(R5)
01 10 0033 509 BSBB CNX$CONNECT ; Request connection
05 0037 510 RSB ; Unable to allocate memory
```



```
0038 512      .SBTTL CNX$CONNECT - Connect to remote system
0038 513
0038 514      :++
0038 515
0038 516      FUNCTIONAL DESCRIPTION:
0038 517
0038 518      This routine is called to initiate a connection to the
0038 519      connection manager on a remote system.
0038 520
0038 521      CALLING SEQUENCE:
0038 522
0038 523      JSB      CNX$CONNECT
0038 524      IPL must be at IPL$_SYNCH
0038 525
0038 526      INPUT PARAMETERS:
0038 527
0038 528      R5 is address of initialized CSB
0038 529
0038 530      OUTPUT PARAMETERS:
0038 531
0038 532      NONE
0038 533
0038 534      COMPLETION CODES:
0038 535
0038 536      NONE
0038 537
0038 538      SIDE EFFECTS:
0038 539
0038 540      R0-R5 are destroyed
0038 541
0038 542      :--
0038 543
0038 544      CNX$CONNECT::
0038 545      :
0038 546      : Try to connect
0038 547      : This thread may be suspended here
0038 548      :
0038 549      BSBW      CNCT DATA      ; Set up connect data
0038 550      MOVL      CSB$[ SB(R5),R0 ; Address of SB
0038 551      CONNECT    MSGADR = CNX$RCV_MSG,- ; Connect to system
0038 552      ERRADR = CNX$ERROR,-
0038 553      LPRNAM = PROC_NAME,-
0038 554      RPRNAM = PROC_NAME,-
0038 555      RSYID = SB$B_SYSTEMID(R0),-
0038 556      INITCR = #SEND_CREDITS,-
0038 557      CONDAT = CSB$B_CNCT(R5),-
0038 558      AUXSTR = (R5)
0038 559      BLBS      R0,5$
0038 560      BSBW      CNX_STATUS_CHECK ; Check for bugcheck request
0038 561      5$:
0038 562      STATE_DISP <<CONNECT,100$>,<RECONNECT,200$>,<DEAD,300$>>
0038 563      10$: BLBC      R0,20$ ; OK if connect failed
0038 564      MOVZWL    #<CLMDRSSM_DRS ! - ; Disconnect status
0038 565      CLMDRSSC_PROTOCOL>,R0
0038 566      15$: DISCONNECT ; Break anomalous connection
0038 567      20$: RSB ; Drop thread
0038 568
```

50 03C4 30 0038 549
68 A5 D0 003B 550
003F 551
003F 552
003F 553
003F 554
003F 555
003F 556
003F 557
003F 558
03 50 E8 0072 559
04DE 30 0075 560
0078 561
0078 562
50 0B 50 E9 0085 563
8004 8F 3C 0088 564
008D 565
008D 566
05 0093 567
0094 568

```
50 8104 8F 3C 0094 569 90$: MOVZWL #<CLMDRSSM_DRS ! - : Disconnect status
0099 570 : CLMDRSSM-FATAL ! - : Bugcheck request
0099 571 : CLMDRSSC_PROTOCOL>,R0
F2 11 0099 572 BRB 15$
009B 573
009B 574
009B 575 : Initial connect attempt completed
009B 576
009B 577 R2 is address of connection message
009B 578 R3 is address of CDT
009B 579 R4 is address of PDT
009B 580 R5 is address of CSB
009B 581
43 50 E9 009B 582 100$: BLBC R0,150$ : Connection error
009E 583 ASSUME CNCT$B_VERNUM EQ CNCT$B_ECOLVL+1
009E 584 ASSUME CSB$B_VERNUM EQ CSB$B_ECOLVL+1
20 A2 B0 009E 585 MOVW SCSCMSG$B_SNDDAT+CNCT$B_ECOLVL(R2) - : Store remote side's
40 A5 80 00A1 586 CSB$B_ECOLVL(R5) : protocol version number and ECO level
23 A2 90 00A3 587 MOVW SCSCMSG$B_SNDDAT+CNCT$B_ACKLIM(R2) - : Store remote side's
33 A5 00A6 588 CSB$B_REACKLIM(R5) : ACK limit
037F 30 00A8 589 BSBW CNCT_CHECK : Check connect data
E6 50 E9 00AB 590 BLBC R0,90$ : Bugcheck remote node
00AE 591 ASSUME CSB$B_PDT EQ CSB$B_CDT+4
OC A5 53 7D 00AE 592 MOVW R3,CSB$B_CDT(R5) : Store CDT and PDT address
01 90 00B2 593 MOVW #CSB$B_OPEN - : Mark connection open
43 A5 00B4 594 CSB$B_STATE(R5)
52 68 A5 DD 00B6 595 PUSHL R2 : Save connection message addr
52 18 A2 DO 00B8 596 MOVL CSB$B_SB(R5),R2 : Address of System Block
51 64 A5 DO 00BC 597 MOVAB SB$B_SYSTEMID(R2),R2 : Address of destination system ID
51 OC A1 DO 00C0 598 MOVL CSB$B_CLUB(R5),R1 : Address of cluster block
50 D4 00C8 599 MOVL CLUB$B_POLL_CTX(R1),R1 : Address of SPPB
00000000 GF 16 00CA 600 CLRL R0 : Disable polling
04 BA 00D0 601 JSB G^SCSPOLL_MODE : Disable polling this system
50 0000 CF 9E 00D2 602 POPR #^M<R2> : Restore connection message addr
FF26 30 00D7 603 MOVAB CNCT_MSG,R0 : Connect message address
FF23 30 00DA 604 BSBW CNX$CONFIG_CHANGE : Note configuration change
FF20 30 00DD 605 BSBW CNX$RESEND_MSGS : Initialize connection
05 00E0 606 BSBW CNX$CON_NEWSYS : Inform configuration manager of new system
00E1 607 RSB : Return
00E1 608
00E1 609 150$: : Come here on failure to make a connection.
0424 30 00E1 610 BSBW CNX$DECREFCNT : Deallocate CSB and return
05 00E4 611 RSB
00E5 612
00E5 613
00E5 614 : Reconnect completed
00E5 615
00E5 616
3A 50 E9 00E5 617 200$: BLBC R0,220$
00E8 618 ASSUME CSB$B_PDT EQ CSB$B_CDT+4
OC A5 53 7D 00E8 619 MOVW R3,CSB$B_CDT(R5) : Store CDT and PDT address
0389 30 00EC 620 BSBW RECNCT_CHECK : Check data following reconnect
59 50 E9 00EF 621 BLBC R0,280$ : Other node should crash
4D 60 A5 00 00F2 622 BBS #CSB$B_LONG_BREAK - : Disconnect if long break
23 A2 90 00F7 623 CSB$B_STATUS(R5),270$ : has been seen
33 A5 00FA 624 MOVW SCSCMSG$B_SNDDAT+CNCT$B_ACKLIM(R2) - : Store remote side's
33 A5 00FA 625 CSB$B_REACKLIM(R5) : ACK limit
```

```

      D1 90 00FC 626      MOVB #CSB$K_OPEN,-      ; Mark connection open
      43 A5 00FE 627      CSB$B_STATE(R5)
      FEFD' 30 0100 628      BSBW CNX$CHECK_QUORUM      ; Resume activity on if quorum
      50 0000'CF 9E 0103 629      MOVAB RECNET_MSG,R0      ; Address of reconnect message
      FEF5' 30 0108 630      BSBW CNX$CONFIG_CHANGE      ; Note configuration change
      50 2C A2 B0 010B 631      MOVW -      ; Get last received sequence number
      50 30 A5 A3 010F 632      SCSCMSG$B_SNDDAT+CNCT$W_RCVDSQNM(R2),R0
      51 0113 633      SUBW3 CSB$W_ACRSEQNM(R5),-      ; Is it .ge. last one?
      08 19 0114 634      R0,R1
      30 A5 50 B0 0116 635      BLSS 210$      ; Branch if not and bugcheck
      FEE3' 30 011A 636      MOVW R0,CSB$W_ACRSEQNM(R5)      ; Store updated number
      05 011D 637      BSBW CNX$RESEND_MSGS      ; Send pending message, initialize connectio
      011E 638      RSB
      0122 639      210$: BUG_CHECK CNXMGRERR,FATAL      ; Invalid acknowledged sequence number
      0D 60 A5 00 E0 0122 641      220$: BBS #CSB$V_LONG_BREAK,-      ; Branch if long break has already
      48 A5 00000000'GF D1 0127 642      CSB$L_STATUS(R5),240$      ; been seen
      0A 1F 0127 643      CMPL G^EXESGL_ABSTIM,-      ; Have we retried for long enough?
      03BA 30 012F 644      CSB$L_TIMEOUT(R5)
      54 2710 8F 3C 0131 645      BLSSU 250$      ; Not timeout out yet
      05 11 0131 646      BSBW LONG_BREAK      ; Long break seen
      54 03E8 8F 3C 0134 647      MOVZWL #10000,R4      ; 10 sec = 10000 ms timeout
      05 11 0139 648      BRB 260$
      54 03E8 8F 3C 0138 649      240$: MOVZWL #1000,R4      ; 1 sec = 1000 ms
      0227 30 0138 651      BSBW CNX$WAIT      ; Start timeout
      05 0140 652      RSB
      50 8004 8F 3C 0143 653      250$: MOVZWL #<CLMDRSS$M_DRS ! -      ; Disconnect status
      05 11 0144 654      CLMDRSS$C_PROTOCOL>,R0
      05 11 0149 655      BRB 290$      ; Branch to common code
      3C 0148 656      280$: MOVZWL #< -      ; Other node should withdraw
      014C 657      CLMDRSS$M_DRS ! -
      014C 658      CLMDRSS$C_REMOVED -
      50 800A 8F 30 014C 659      >, R0
      01DE 30 0150 660      290$: BSBW CNX$DISCONNECT      ; Break connection
      05 0153 661      RSB
      FF2E 30 0154 662      300$: BSBW 10$      ; Drop connection
      03F5 30 0157 663      BSBW DEAD_NODE      ; Make node die
      05 015A 664      RSB      ; Return
      665
      666
      667
      668
```



```
0158 670 .SBTTL CNX$RCV_CNCT_MSG - Receive CONNECT message
0158 671
0158 672 :++
0158 673
0158 674 : FUNCTIONAL DESCRIPTION:
0158 675
0158 676 This routine is called by SCS when a incoming CONNECT occurs
0158 677 for us. First the list of CSBs is scanned to see if we had a
0158 678 previous connection to that system. If we did then if the
0158 679 software incarnation is the same we have to resend any messages
0158 680 that haven't been received. If the software incarnation changed,
0158 681 then we have to do a failover. If we don't have a CSB for that
0158 682 system then one is created.
0158 683
0158 684 : CALLING SEQUENCE:
0158 685
0158 686 JSB CNX$RCV_CNCT_MSG
0158 687
0158 688 : INPUT PARAMETERS:
0158 689
0158 690 R2 Address of connect request message
0158 691 R3 Address of CDT
0158 692 R4 Address of PDT
0158 693
0158 694 : IMPLICIT INPUTS:
0158 695
0158 696 None
0158 697
0158 698 : OUTPUT PARAMETERS:
0158 699
0158 700 None
0158 701
0158 702 : IMPLICIT OUTPUTS:
0158 703
0158 704 Completion codes returned to remote system if connection
0158 705 is rejected:
0158 706
0158 707 $$$_REJECT Connection rejected. R1 is in CLMDRS format.
0158 708 $$$_INSFMEM Unable to allocate memory
0158 709
0158 710 : COMPLETION CODES:
0158 711
0158 712 None
0158 713
0158 714 : SIDE EFFECTS:
0158 715
0158 716 None
0158 717
0158 718 :--
0158 719
0158 720 CNX$RCV_CNCT_MSG:
51 1C A3 D0 0158 721 10$: MOVL CDT$ PB(R3),R1 ; Get address of path block
51 30 A1 D0 0158 722 MOVL PB$ SBLINK(R1),R1 ; Get address of system block
0430 30 0163 723 BSBW CNX$LOOKUP_CSB ; Find CSB
1B 50 E9 0166 724 BLBC R0,30$ ; Branch on error
0169 725 STATE DISP <<NEW,200$>,<CONNECT,100$>,<RECONNECT,300$>,<WAIT,400$>>
0179 726 BUG_CHECK CNXMGRERR,FATAL ; Unexpected connect received
```

```
017D 727 : MOVZWL #<CLMDRSSM_DRS ! - : Protocol error
017D 728 : CLMDRSSC_PROTOCOL ! -
017D 729 : CLMDRSSM_FATAL>,R0
017D 730 : BRB 30$
017D 731 :
50 8006 BF 3C 017D 732 20$: MOVZWL #<CLMDRSSM_DRS ! - : Protocol version error
0182 733 : CLMDRSSC_VERSION>,R0
OC 11 0182 734 : BRB 30$
0184 735 :
50 8004 BF 3C 0184 736 30$: MOVZWL #<CLMDRSSM_DRS ! - : Nonfatal protocol error
0189 737 : CLMDRSSC_PROTOCOL>,R0
05 11 0189 738 : BRB 30$
018B 739 :
50 8104 BF 3C 018B 740 40$: MOVZWL #<CLMDRSSM_DRS ! - : Fatal protocol error
0190 741 : CLMDRSSM_FATAL ! -
0190 742 : CLMDRSSC_PROTOCOL>,R0
0190 743 50$: REJECT : Reject the connection
05 0193 744 RSB
0194 745 :
0194 746 :
0194 747 : Connect request from system to which we are connecting
0194 748 : Compare system ids and allow the system with the lower id to CONNECT
0194 749 : and the one with the higher id to ACCEPT.
0194 750 :
00000018'GF 52 68 OC BB 0194 751 100$: PUSHR #*M<R2,R3> : Save R2 and R3 from CMPC3
18 A2 A5 D0 0196 752 : MOVL CSB$S_SB(R5),R2 : Address of System Block
OC 06 29 019A 753 : CMPC3 #SB$S_SYSTEMID, - : Compare system ids
01A3 754 : SB$S_SYSTEMID(R2), -
01A3 755 : G*SC$S$GA_LOCALSB+SB$S_SYSTEMID
OC BA 01A3 756 : POPR #*M<R2,R3> : Restore R2 and R3
DD 1E 01A5 757 : BGEQU 30$ : Remote is higher - reject
01A7 758 :
01A7 759 : Connect request -- expected to be from a new system.
01A7 760 : If the connect request is not from a newly booting system, we assume
01A7 761 : that the local system has booted twice with the same incarnation
01A7 762 : number. In this case, we reject the connection and hope that the other
01A7 763 : node will eventually see our attempts to connect and realize that this
01A7 764 : node has duplicated its incarnation number.
01A7 765 :
22 A2 91 01A7 766 200$: CMPB SCSCMG$B_SNDDAT+CNCT$B_TYPE(R2).- : Verify remote side is doing
01 01AA 767 : #CNCT$K_INITIAL : a new CONNECT
D7 12 01AB 768 : BNEQ 30$ : It's not - don't allow a connect
01AD 769 : ASSUME CNCT$B_VERNUM EQ CNCT$B_ECOLVL+1
01AD 770 : ASSUME CSB$B_VERNUM EQ CSB$B_ECOLVL+1
20 A2 B0 01AD 771 : MOVW SCSCMG$B_SNDDAT+CNCT$B_ECOLVL(R2).- : Store remote side's
40 A5 01B0 772 : CSB$B_ECOLVL(R5) : protocol version number and ECO level
23 A2 90 01B2 773 : MOVB SCSCMG$B_SNDDAT+CNCT$B_ACKLIM(R2).- : Store remote side's
33 A5 01B5 774 : CSB$B_REACKLIM(R5) : ACK limit
0270 30 01B7 775 : BSBW CNCT_CHECK : Check connect data
CE 50 E9 01BA 776 : BLBC R0,40$ : Branch to bugcheck remote node
06 90 01BD 777 : MOVB #CSB$K_ACCEPT,- : Set state
43 A5 01BF 778 : CSB$B_STATE(R5)
SF 10 01C1 779 : BSBW CNX$ACCEPT : Accept connection
05 01C3 780 : RSB
01C4 781 :
01C4 782 :
01C4 783 : Reconnect request from a system to which we are reconnecting
```

```
00000018'GF 52 68 0C BB 01C4 784
18 A2 06 29 01C4 785 300$: PUSHB #M<R2,R3> ; Save R2 and R3 from CMPC3
01C6 786 ; CSBSL_SB(R5),R2 ; Address of System Block
01CA 787 ; #CSBS-SYSTEMID,- ; Compare system ids
01D3 788 ; SB$B_SYSTEMID(R2),-
01D3 789 ; G^SC$GA_LOCALSB+SB$B_SYSTEMID
01D3 790 ; POPR #M<R2,R3> ; Restore R2 and R3
AD 1E 01D5 791 ; BGEQU 30$ ; Remote is higher - reject
01D7 792
01D7 793 : Connect request from a system that we are timing out
01D7 794
22 A2 91 01D7 795 400$: CMPB SCSCMSG$B_SNDDAT+CNCT$B_TYPE(R2),- ; Verify remote side is doing
02 01DA 796 ; #CNCT$K_RECONNECT ; a RECONNECT
35 12 01DB 797 ; BNEQ 430$ ; It's not -- handle special situation
0298 30 01DD 798 ; BSBW RECNT_CHECK ; Check data following reconnect
9F 60 A5 00 E0 01E0 799 ; BBS #CSBSV_LONG_BREAK,- ; Reject if long break has been
01E5 800 ; CSBSL_STATUS(R5),30$
01E5 801 ; ASSUME CNCT$B_VERNUM EQ CNCT$B_ECOLVL+1
01E5 802 ; ASSUME CSBS$B_VERNUM EQ CSBS$B_ECOLVL+1
20 A2 B1 01E5 803 ; CMPW SCSCMSG$B_SNDDAT+CNCT$B_ECOLVL(R2),- ; Are we speaking the
40 A5 01E8 804 ; CSBS$B_ECOLVL(R5) ; same protocol as before?
91 12 01EA 805 ; BNEQ 20$ ; No -- fatal error
23 A2 90 01EC 806 ; MOVW SCSCMSG$B_SNDDAT+CNCT$B_ACKLIM(R2),- ; Store remote side's
33 A5 01EF 807 ; CSBS$B_REACKLIM(R5) ; ACK limit
08 90 01F1 808 ; MOVW #CSBS$K_REACCEPT,- ; Set state
43 A5 01F3 809 ; CSBS$B_STATE(R5)
2C A2 B0 01F5 810 ; MOVW SCSCMSG$B_SNDDAT+CNCT$B_RCVSEQNM(R2),- ; Get last received
50 01F8 811 ; R0 ; sequence number (of ours)
30 A5 A3 01F9 812 ; SUBW3 CSBS$B_ACKRSEQNM(R5),- ; Verify it's greater than or equal to
51 50 01FC 813 ; R0,R1 ; the last one
04 18 01FE 814 ; BGEQ 420$ ; It is
0200 815 ; BUG_CHECK CNXMGRERR,FATAL ; *** Sequence number error
0204 816
30 A5 50 B0 0204 817 420$: MOVW R0,CSBS$B_ACKRSEQNM(R5) ; It's ok - store it
0320 30 0208 818 ; BSBW DELETE TQE ; Clean out TQE
7E A5 02 90 020B 819 ; MOVW #CNCT$K_RECONNECT,- ; Flag this as a reconnect
02CF 820 ; CSBS$B_CNCT+CNCT$B_TYPE(R5)
11 10 020F 821 ; BSBW CNX$ACCEPT ; Accept connection
05 0211 822 ; RSB
0212 823
0212 824
0212 825 : Get here is this node expected a re-connection and instead received
0212 826 ; an initial connect request. This happens if the remote node has rebooted
0212 827 ; with a duplicated software incarnation number. To recover from this
0212 828 ; situation, the software incarnation number in the CSB is modified and
0212 829 ; this routine is re-entered. The old CSB will be marked 'DEAD'.
0212 830 ; A new CSB will be formed and the connect request accepted.
0212 831
50 0323 30 0212 832 430$: BSBW DELETE TQE ; Clean out TQE
38 A5 7E 0215 833 ; MOVW CSBS$B_SWINCARN(R5),R0 ; Address of stored software incarnation num
80 60 D2 0219 834 ; MCOML (R0),(R0)+ ; Invalidate software incarnation number
60 60 D2 021C 835 ; MCOML (R0),(R0) ; so that a new CSB will be formed
FF39 31 021F 836 ; BRW 10$ ; Re-enter this routine
```



```
0222 838 .SBTTL CNXSACCEPT - Accept connection from remote system
0222 839
0222 840 :++
0222 841 :
0222 842 : FUNCTIONAL DESCRIPTION:
0222 843 :
0222 844 : This routine is called to accept a connection from the
0222 845 : connection manager on a remote system.
0222 846 :
0222 847 : CALLING SEQUENCE:
0222 848 :
0222 849 : JSB CNXSACCEPT
0222 850 : IPL must be at IPL$_SYNCH
0222 851 :
0222 852 : INPUT PARAMETERS:
0222 853 :
0222 854 : R2 is address of connection message
0222 855 : R3 is address of CDT
0222 856 : R4 is address of PDT
0222 857 : R5 is address of initialized CSB
0222 858 :
0222 859 : OUTPUT PARAMETERS:
0222 860 :
0222 861 : NONE
0222 862 :
0222 863 : COMPLETION CODES:
0222 864 :
0222 865 : NONE
0222 866 :
0222 867 : SIDE EFFECTS:
0222 868 :
0222 869 : R0-R5 are destroyed
0222 870 :
0222 871 :--
0222 872 :
0222 873 : CNXSACCEPT::
0222 874 :
0222 875 : Try to accept
0222 876 : This thread may be suspended here
0222 877 :
01DA 30 0222 878 BSBW CNCT DATA : Setup connect data
0222 879 ACCEPT MSGADR = CNX$RCV MSG,-
0222 880 ERRADR = CNX$ERROR,-
0222 881 INITCR = #SEND_CREDITS,-
0222 882 CONDAT = CSB$B_CNCT(R5),-
0222 883 AUXSTR = (R5)
06 50 E6 0222 884 BLBS R0,10$ : Branch on success
3F BB 0222 885 PUSHF #^M<R0,R1,R2,R3,R4,R5> : Save registers
10 10 0222 886 BSBB 20$ : Clean up failed success
3F BA 0222 887 POPR #^M<R0,R1,R2,R3,R4,R5> : Restore registers
0222 888 10$:
0222 889 STATE DISP <<ACCEPT,100$>,<REACCEPT,200$>>
0222 890 BUG_CHECK CNXMGRERR,FATAL ; Bugcheck
0222 891 :
0222 892 :
0222 893 : Accept attempt failed.
0222 894 : Clean up by rejecting a connection based on the listening CDT.
```

```

      53  52  D0  0262  895  :
      05      0262  896  20$:  MOVL    R2,R3          : Address of Listening CDT
      05      0265  897          REJECT          : Reject the request
      05      0268  898          RSB              : Terminate the thread
      05      0269  899
      05      0269  900
      05      0269  901  : Initial accept attempt completed
      05      0269  902
      05      0269  903          R3 is address of CDT
      05      0269  904          R4 is address of PDT
      05      0269  905          R5 is address of CSB
      05      0269  906
      2F  50  E9  0269  907  100$:  BLBC    R0,150$          : ACCEPT error
      0C  A5  53  7D  026C  908          ASSUME  CSB$PDT EQ CSB$PDT+4
      52  68  A5  D0  0270  909          MOVQ    R3,CSB$PDT(R5)      : Store CDT and PDT address
      52  18  A2  9E  0274  910          MOVL    CSB$SBTR5),R2      : Address of System Block
      51  64  A5  D0  0278  911          MOVAB   SB$SYSTEMID(R2),R2  : Address of destination system ID
      51  0C  A1  D0  027C  912          MOVL    CSB$CLUB(R5),R1      : Address of CLUB
      00000000'GF  16  0282  913          MOVL    CLUB$_POLL_CTX(R1),R1 : Address of SPPB
      05      0280  914          CLRL    R0              : Disable polling
      05      0282  915          JSB     G$SCSPOLL MODE      : Disable polling this system
      05      0288  916          MOVB    #CSB$K_OPEN,-      : Mark connection open
      05      028A  917          CSB$STATE(R5)
      50  0000'CF  9E  028C  918          MOVAB   ACPT_MSG,R0      : Address of accept message
      05      0291  919          BSBW    CNX$CONFIG_CHANGE      : Note configuration change
      05      0294  920          BSBW    CNX$RESEND_MSGS      : Initialize for sending messages
      05      0297  921          BSBW    CNX$CON_NEWSYS      : Inform configuration manager
      05      029A  922          RSB
      05      029B  923
      026A  30  029B  924  150$:  BSBW    CNX$DECREFCNT      : Deallocate CSB
      05      029E  925          RSB
      05      029F  926
      05      029F  927  : Reaccept attempt completed
      05      029F  928
      05      029F  929
      05      029F  930          R3 is address of CDT
      05      029F  931          R4 is address of PDT
      05      029F  932          R5 is address of CSB
      05      029F  933
      1C  50  E9  029F  934  200$:  BLBC    R0,210$          : Branch on failure
      0C  A5  53  7D  02A2  935          ASSUME  CSB$PDT EQ CSB$PDT+4
      1C  60  A5  00  E0  02A6  936          MOVQ    R3,CSB$PDT(R5)      : Store CDT and PDT address
      05      02AB  937          BBS     #CSB$V_LONG_BREAK,-      : Branch if long break in connection
      05      02AB  938          CSB$STATUS(R5),220$
      05      02AB  939          MOVB    #CSB$K_OPEN,-      : Mark connection open
      05      02AD  940          CSB$STATE(R5)
      05      02AF  941          BSBW    CNX$CHECK_QUORUM      : Resume activity on if quorum
      50  0000'CF  9E  02B2  942          MOVAB   REACPT_MSG,R0      : Address of reaccept message
      05      02B7  943          BSBW    CNX$CONFIG_CHANGE      : Note configuration change
      05      02BA  944          BSBW    CNX$RESEND_MSGS      : Resend outstanding messages
      05      02BD  945          RSB
      05      02BE  946
      54  03E8  8F  3C  02BE  947  210$:  MOVZWL  #1000,R4      : Wait 1 sec = 1000 ms
      05      02C3  948          BSBW    CNX$WAIT          : Enter wait state
      05      02C6  949          RSB
      05      02C7  950
      50  800A  8F  3C  02C7  951  220$:  MOVZWL  #<CLMDRSSM_DRS ! -
```

CNXMAN
V04-000

- Cluster Connection Manager
CNXSACCEPT - Accept connection from remo

L 9

16-SEP-1984 00:24:50 VAX/VMS Macro V04-00
5-SEP-1984 04:07:15 [SYSLOA.SRC]CNXMAN.MAR;1

Page 20
(9)

63	10	02CC	952		CLMDRSSC REMOVED>,RO	
	05	02CC	953	BSBB	CNX\$DISCONNECT	: Break connection
		02CE	954	RSB		


```
02CF 956 .SBTTL CNX$DISC_BUGCHECK - Disconnect from Node and Request it to Bugcheck
02CF 957 .SBTTL CNX$DISC_REMOVE - Disconnect from Node Removed from Cluster
02CF 958 .SBTTL CNX$DISC_PROTOCOL - Disconnect from Node for protocol reasons
02CF 959 :++
02CF 960 :
02CF 961 FUNCTIONAL DESCRIPTION:
02CF 962 :
02CF 963 These routine are called to break a connection with a remote node and
02CF 964 return some informational status.
02CF 965 :
02CF 966 CNX$DISC_BUGCHECK: Request the remote node to bugcheck
02CF 967 CNX$DISC_REMOVE: Status indicates that the node was removed from the
02CF 968 cluster
02CF 969 CNX$DISC_PROTOCOL: Disconnect for protocol reasons, reconnect as convenient
02CF 970 :
02CF 971 CALLING SEQUENCE:
02CF 972 :
02CF 973 JSB CNX$DISC_BUGCHECK
02CF 974 JSB CNX$DISC_REMOVE
02CF 975 JSB CNX$DISC_PROTOCOL
02CF 976 IPL is at SCS fork level
02CF 977 :
02CF 978 INPUT PARAMETERS:
02CF 979 :
02CF 980 R5: Address of CSB of removed node
02CF 981 :
02CF 982 IMPLICIT INPUTS:
02CF 983 :
02CF 984 None
02CF 985 :
02CF 986 OUTPUT PARAMETERS:
02CF 987 :
02CF 988 None
02CF 989 :
02CF 990 IMPLICIT OUTPUTS:
02CF 991 :
02CF 992 None
02CF 993 :
02CF 994 SIDE EFFECTS:
02CF 995 :
02CF 996 R0-R1 destroyed
02CF 997 :
02CF 998 :--
02CF 999 :
02CF 1000 .ENABLE LSB
02CF 1001 :
02CF 1002 CNX$DISC_BUGCHECK::
50 8104 8F 3C 02CF 1003 MOVZWL #CLMDRSSM_DRS ! - ; Disconnect status
02D4 1004 CLMDRSSM_FATAL ! - ; Fatal
02D4 1005 CLMDRSSC_PROTOCOL, R0
02D6 1006 BRB 10$ ; Join common code
02D6 1007 :
02D6 1008 CNX$DISC_REMOVE::
50 800A 8F 3C 02D6 1009 MOVZWL #CLMDRSSM_DRS ! - ; Disconnect status
02DB 1010 CLMDRSSC_REMOVED, R0
09 10 02DB 1011 10$: BSBB DISC_STATUS ; Disconnect using status in R0
020E 30 02DD 1012 BSBB LONG_BREAK ; Mark a long break (may already be done)
```

```

05 02E0 1013 RSB ; Return to caller
02E1 1014
02E1 1015 CNX$DISC_PROTOCOL::
50 8004 8F 3C 02E1 1016 MOVZWL #CLMDRSSM_DRS ! - ; Disconnect status
02E6 1017 CLMDRSSC_PROTOCOL, R0
02E6 1018 DISC_STATUS: ; Disconnect status in R0
3C BB 02E6 1019 PUSHF #^M<R2,R3,R4,R5> ; Save non-volatile registers
02E8 1020 STATE_DISP <<OPEN,100$>> ; Connection is currently open
02 11 02EF 1021 BRB 190$
02F1 1022
16 10 02F1 1023 100$: BSBB CNX$BREAK ; Break connection, status in R0
3C BA 02F3 1024 190$: POPR #^M<R2,R3,R4,R5> ; Restore registers
05 02F5 1025 RSB
02F6 1026
02F6 1027 .DISABLE LSB
```

```
02F6 1029      .SBTTL CNX$ERROR - Connection error
02F6 1030
02F6 1031      :++
02F6 1032
02F6 1033      FUNCTIONAL DESCRIPTION:
02F6 1034
02F6 1035          This routine is called by SCS when a connection breaks.
02F6 1036          This routine calls CNX$CLEANUP to deal with outstanding
02F6 1037          messages and then does a DISCONNECT. A timeout is then
02F6 1038          requested at the conclusion of which the connection will
02F6 1039          be reattempted.
02F6 1040
02F6 1041      CALLING SEQUENCE:
02F6 1042
02F6 1043          JSB      CNX$ERROR (Called from SCS)
02F6 1044          IPL is at SCS fork level (8)
02F6 1045
02F6 1046      INPUT PARAMETERS:
02F6 1047
02F6 1048          R0      Contains error status (SS$_DISCONNECT or SS$_VCBROKEN)
02F6 1049          R1      Additional status (disconnect reason or virtual circuit
02F6 1050                  broken reason)
02F6 1051          R3      Address of CDT
02F6 1052          R4      Address of PDT
02F6 1053
02F6 1054      IMPLICIT INPUTS:
02F6 1055
02F6 1056          None
02F6 1057
02F6 1058      OUTPUT PARAMETERS:
02F6 1059
02F6 1060          None
02F6 1061
02F6 1062      IMPLICIT OUTPUTS:
02F6 1063
02F6 1064          None
02F6 1065
02F6 1066      SIDE EFFECTS:
02F6 1067
02F6 1068          R0-R5 destroyed
02F6 1069
02F6 1070      :--
02F6 1071
02F6 1072      CNX$ERROR::
02F6 1073          MOVL     CDT$AUXSTRUC(R3),R5      ; CSB address
02FA 1074          PUSHR    #^M<R0,R1,R5>      ; Save registers
02FC 1075          MOVZWL   #CLMDRSSM.DRS ! -    ; Disconnect status
0301 1076          CLMDRSSC PROTOCOL,R0
0301 1077          BSBB      CNX$BREAK                ; Use common disconnect code
0303 1078          POPR     #^M<R0,R1,R5>      ; Restore registers
0305 1079          BSBW     CNX_STATUS_CHECK        ; Check for bugcheck request
0308 1080          RSB
```

```
55  5C  A3  D0  02F6 1073
      23  BB  02FA 1074
50  8004 8F  3C  02FC 1075
      06  10  0301 1076
      23  BA  0303 1077
      024E 30  0305 1078
      05  0308 1080
```


.SBTTL CNX\$BREAK - Cleanup and Disconnect SCS Connection

0309 1082
0309 1083
0309 1084
0309 1085
0309 1086
0309 1087
0309 1088
0309 1089
0309 1090
0309 1091
0309 1092
0309 1093
0309 1094
0309 1095
0309 1096
0309 1097
0309 1098
0309 1099
0309 1100
0309 1101
0309 1102
0309 1103
0309 1104
0309 1105
0309 1106
0309 1107
0309 1108
0309 1109
0309 1110
0309 1111
0309 1112
0309 1113
0309 1114
0309 1115
0309 1116
0309 1117
0309 1118
0309 1119
0309 1120
0309 1121
0309 1122
0309 1123
0309 1124
0310 1125
0310 1126
0317 1127
0320 1128
0320 1129
0325 1130
0328 1131
0328 1132
032A 1133
032A 1134
032E 1135
0330 1136

:++

FUNCTIONAL DESCRIPTION:

This routine is called when a connection breaks or when a connection is to be broken. It calls CNX\$CLEANUP to deal with outstanding messages and then does a DISCONNECT. A timeout is then requested at the conclusion of which the connection will be reattempted.

CALLING SEQUENCE:

JSB CNX\$BREAK
IPL is at IPL\$_SYNCH = IPL\$_SCS

INPUT PARAMETERS:

R0 Contains disconnect code
R5 Address of CSB

IMPLICIT INPUTS:

None

OUTPUT PARAMETERS:

None

IMPLICIT OUTPUTS:

None

SIDE EFFECTS:

R0-R4 Destroyed.

:--

CNX\$BREAK::

PUSHL R0 ; Save disconnect status
BBS ; Branch if long break
MOVZWL G^CLUS\$GW RECNXINT, R0 ; Max. retry in seconds
ADDL3 R0, G^EXE\$GL ABSTIM, - ; Time at which to stop retries
CSB\$L TIMEOUT(R5)
MOVAB CNXERROR MSG, R0 ; Address of message
BSBW CNX\$CONFIG_CHANGE ; Note configuration change
20\$: POPR #^M<R0> ; Restore disconnect status
ASSUME CSB\$L PDT EQ CSB\$L_CDT+4 ; Fetch CDT and PDT addresses
MOVQ CSB\$L_CDT(R5), R3
BSBB CNX\$DISCONNECT ; Disconnect, status in R0
RSB

18 60 A5 50 DD 0309 1123
00 00 E0 030B 1124
50 00000000'GF 3C 0310 1125
48 A5 00000000'GF 50 C1 0317 1127
50 0000'CF 9E 0320 1128
FCD8' 30 0320 1129
01 BA 0325 1130
53 0C A5 7D 0328 1132
01 10 032A 1133
05 032E 1134
0330 1135
0330 1136

```
0331 1138 .SBTTL CNX$DISCONNECT - Disconnect from remote system
0331 1139
0331 1140 :++
0331 1141
0331 1142 FUNCTIONAL DESCRIPTION:
0331 1143
0331 1144 This routine is called to d'sconnect from the
0331 1145 connection manager on a remote system.
0331 1146
0331 1147 CALLING SEQUENCE:
0331 1148
0331 1149 JSB CNX$DISCONNECT
0331 1150 IPL must be at IPL$_SYNCH
0331 1151
0331 1152 INPUT PARAMETERS:
0331 1153
0331 1154 R0 is disconnect status
0331 1155 R5 is address of initialized CSB
0331 1156
0331 1157 OUTPUT PARAMETERS:
0331 1158
0331 1159 NONE
0331 1160
0331 1161 COMPLETION CODES:
0331 1162
0331 1163 NONE
0331 1164
0331 1165 SIDE EFFECTS:
0331 1166
0331 1167 R0-R5 are destroyed
0331 1168
0331 1169 :--
0331 1170
0331 1171 CNX$DISCONNECT::
0331 1172 MOVB #CSB$K_DISCONNECT, - ; Set disconnect state
0331 1173 CSB$B_STATE(R5)
0331 1174 PUSHL R0 ; Save status
0331 1175 BSBW CNX$CHECK_QUORUM ; Block activity if quorum lost
0331 1176 BSBW CNX$PRE_CLEANUP ; Cleanup outstanding messages
0331 1177 POPR #*M<R0> ; Restore status
0331 1178
0331 1179 : Try to disconnect
0331 1180 : This thread may be suspended here
0331 1181 :
0331 1182 ASSUME CSB$PDT EQ CSB$PDT+4
0331 1183 MOVQ CSB$PDT(R5),R3 ; Fetch CDT and PDT addresses
0331 1184 DISCONNECT ; Status in R0, always succeeds
0331 1185 STATE_DISP <<DISCONNECT,100$>>
0331 1186 BUG_CHECK CNXMGRERR,FATAL ; Invalid state
0331 1187
0331 1188 :
0331 1189 : Soft disconnect attempt completed
0331 1190
0331 1191 100$: BSBW CNX$POST_CLEANUP ; Finish cleanup of outstanding messages
0331 1192 ASSUME CSB$PDT EQ CSB$PDT+4
0331 1193 CLRQ CSB$PDT(R5) ; Clear CDT and PDT address in CSB
0331 1194 BBC #CSB$V_LONG_BREAK, - ; Branch if no long break yet
```

43 A5 07 90
50 DD
FCC6' 30
FCC3' 30
01 BA

53 OC A5 7D

FCA9' 30

OC A5 7C
03 60 A5 00 E1

54	FC9E'	30	035F	1195		CSB\$L STATUS(R5),110\$	
	01F4	3C	035F	1196		CNX\$FAIL_MSG	: Fail outstanding messages
	8F	10	0362	1197	110\$:	#500,R4	: Delay 500 milli-seconds
	01	05	0367	1198		CNX\$WAIT	
			0369	1199			

BSBW
MOVZWL
BSBB
RSB

036A 1201 .SBTTL CNX\$WAIT - Initiate timeout

036A 1202
036A 1203 :++036A 1204
036A 1205036A 1206
036A 1207036A 1208
036A 1209036A 1210
036A 1211036A 1212
036A 1213036A 1214
036A 1215036A 1216
036A 1217036A 1218
036A 1219036A 1220
036A 1221036A 1222
036A 1223036A 1224
036A 1225036A 1226
036A 1227036A 1228
036A 1229036A 1230
036A 1231036A 1232
036A 1233036A 1234
036A 1235036A 1236
036C 1237036E 1238
0371 12390377 1240
037A 1241037C 1242
0380 12430384 1244
0388 1245038B 1246
038F 12470392 1248
0396 1249039B 1250
03A4 125103AB 1252
03AE 125303B1 1254
03B7 125503BA 1256
03BB 1257

03BB 1257 :++

FUNCTIONAL DESCRIPTION:

This routine is called to begin a timeout before trying
to reconnect to the connection manager on a remote system.

CALLING SEQUENCE:

JSB CNX\$WAIT
IPL must be at IPL\$SYNCH

INPUT PARAMETERS:

R4 is the timeout period in milli-seconds
R5 is address of initialized CSB

OUTPUT PARAMETERS:

NONE

COMPLETION CODES:

NONE

SIDE EFFECTS:

R0-R4 are destroyed

CNX\$WAIT::

```
MOVB #CSB$K WAIT,- ; Set connection state WAITing
CSB$B STATE(R5)
MOVZWL #TQE$K LENGTH,R1 ; Size of timer queue entry
JSB G^EXES$ALONONPAGED ; Allocate one
BLBC R0,RETRY_CONNECT ; No memory, so forget timeout
PUSHL R5 ; Save CSB address
MOVW R1,TQE$W SIZE(R2) ; Store size
MOVB #DYN$C TQE,TQE$B_TYPE(R2) ; Store type
MOVL R5,TQE$L FR3(R2) ; Store CSB address as fork reg. R3
CLRL TQE$L FR4(R2) ; Save zero as fork reg. R4
MOVL R2,CSB$L_TQE(R5) ; Save TQE address in CSB
MOVL R2,R5 ; Move address of TQE
MOVB #TQE$C SSSNGL,TQE$B_RQTYPE(R5) ; Store type of timer queue entry
MOVAB B^TIMEOUT,TQE$L_FPCT(R5) ; Store address of timer fork process
EMUL R4,#10*1000,#0,R3 ; Get milli-seconds and cvt to 100 ns. units
MOVQ G^EXES$GQ_SYSTIME,R0 ; Get current time
ADDL R3,R0 ; Add to current time
ADWC R4,R1
JSB G^EXES$INSTIMQ ; Insert in timer queue
POPL R5 ; Restore CSB address
RSB
```

```
09 90 036A 1235
43 A5 036C 1236
51 30 3C 036E 1237
00000000 GF 16 0371 1238
60 50 E9 0377 1239
55 DD 037A 1240
08 A2 51 B0 037C 1241
0A A2 0F 90 0380 1242
10 A2 55 D0 0384 1243
14 A2 D4 0388 1244
44 A5 52 D0 038B 1245
55 52 D0 038F 1246
08 A5 01 90 0392 1247
OC A5 BB AF 9E 0396 1248
53 7A 039B 1249
50 00000000 GF 7D 03A4 1250
50 53 C0 03AB 1251
51 54 D8 03AE 1252
00000000 GF 16 03B1 1253
55 BED0 03B7 1254
05 03BA 1255
03BB 1256
03BB 1257
```

```
0388 1258 : Come here as a timer fork process to retry the CONNECT
0388 1259 : Inputs:
0388 1260 : R3 CSB address
0388 1261 : R5 TQE address
0388 1262 :
0388 1263 : --
0388 1264 :
0388 1265 : TIMEOUT:
50 55 D0 0388 1266 : MOVL R5,R0 ; Address of timer queue entry
55 55 D0 038E 1267 : MOVL R3,R5 ; Address of CSB
00000000'GF 16 03C1 1268 : JSB G^EXES$DEANONPAGED ; Deallocate it
44 A5 D4 03C7 1269 : CLRL CSB$L_TQE(R5) ; Clear pointer to TQE
43 A5 91 03CA 1270 : CMPB CSB$B_STATE(R5),- ; Are we in wait state?
09 03CD 1271 : #CSB$R_WAIT
02 12 03CE 1272 : BNEQ 10$ ; No, just return
08 10 03D0 1273 : BSBB RETRY_CONNECT ; Do the rest in a subroutine so that
55 00000000'GF DE 03D2 1274 : ; CONNECT can return here
05 03D9 1275 10$: MOVAL G^EXES$L_TQENOREPT,R5 ; Use non-repeating timer queue entry
03DA 1276 : RSB
03DA 1277 :
03DA 1278 :
03DA 1279 : Must check for change to remote system that may have occurred during
03DA 1280 : the timeout. Unlike other situations, there is no connection to break
03DA 1281 : to give notification of such an event.
03DA 1282 :
03DA 1283 : RETRY_CONNECT:
51 68 A5 D0 03DA 1284 : MOVL CSB$L_SB(R5),R1 ; Address of System Block
01B5 30 03DE 1285 : BSBW CNX$LOOKUP_CSB ; Find or allocate a CSB and
0E 50 E9 03E1 1286 : ; as a side effect, handle old CSB
03E1 1287 : BLBC R0,20$ ; Can't allocate CSB
03E4 1288 : STATE_DISP <<NEW,CNX$NEW_CSB>,<WAIT,100$>>
03EE 1289 : BUG_CHECK CNXMGRERR,FATAL
03F2 1290 :
05 03F2 1291 20$: RSB
03F3 1292 :
03F3 1293 100$: MOVB #CSB$K_RECONNECT,- ; Change to RECONNECT state
43 A5 90 03F5 1294 : CSB$B_STATE(R5)
7E A5 02 90 03F7 1295 : MOVB #CNCT$K_RECONNECT,- ; Flag this as a reconnect
FC3A 30 03FB 1296 : CSB$B_CNCT+CNCT$B_TYPE(R5)
05 03FB 1297 : BSBW CNX$CONNECT ; Request connection
03FE 1298 : RSB ; Unable to allocate memory
```

```
03FF 1300 .SBTTL CNCT_DATA - Setup Connect Data in CSB
03FF 1301
03FF 1302 ++
03FF 1303 FUNCTIONAL DESCRIPTION:
03FF 1304 Set up the CSBSB_CNCT area in preparation for requesting
03FF 1305 or accepting a connection.
03FF 1306
03FF 1307 CALLING SEQUENCE:
03FF 1308
03FF 1309 BSBW CNCT_DATA
03FF 1310 IPL must be at IPL$SCS
03FF 1311
03FF 1312 INPUT PARAMETERS:
03FF 1313
03FF 1314 R5 Address of CSB
03FF 1315
03FF 1316 OUTPUT PARAMETERS:
03FF 1317
03FF 1318 None
03FF 1319
03FF 1320 SIDE EFFECTS:
03FF 1321
03FF 1322 R0 and R1 are destroyed.
03FF 1323
03FF 1324
03FF 1325 CNCT_DATA:
03FF 1326 MOVAB CSBSB_CNCT(R5),R0 ; Point to connect data area
0403 1327 MOVL CSBSL_CLUB(R5),R1 ; Address of CLUB
0407 1328 MOVW CLUB$Q_QUORUM(R1), - ; Cluster quorum
040C 1329 CNCT$W_QUORUM(R0)
040C 1330 MOVW CLUB$W_VOTES(R1), - ; Cluster votes
0411 1331 CNCT$W_VOTES(R0)
0411 1332 MOVW CLUB$W_NODES(R1), - ; Cluster nodes
0416 1333 CNCT$W_NODES(R0)
0416 1334 ASSUME CSBSM_LONG_BREAK EQ CNCT$M_LONG_BREAK
0416 1335 ASSUME CSBSM_MEMBER EQ CNCT$M_MEMBER
0416 1336 ASSUME CSBSM_REMOVED EQ CNCT$M_REMOVED
0B A0 60 A5 F8 8F 8B 0416 1337 BICB3 #^C<CSBSM_LONG_BREAK ! -
041D 1338 CSBSM_MEMBER ! -
041D 1339 CSBSM_REMOVED>, -
041D 1340 CSBSL_STATUS(R5), - ; Fill in status bits from CSB
041D 1341 CNCT$B_CNXTS(R0)
041D 1342 ASSUME CLUB$M_CLUSTER EQ CNCT$M_CLUSTER
0A A0 1C A1 FE 8F 8B 041D 1343 BICB3 #^C<CLUB$M_CLUSTER>, -
0424 1344 CLUB$L_FLAGS(R1), - ; Fill in status bits from CLUB
0424 1345 CNCT$B_CLSSTS(R0)
0C A0 2E A5 B0 0424 1346 MOVW CSBSW_RCVDSQNM(R5), - ; Last message received
0429 1347 CNCT$W_RCVDSQNM(R0)
05 0429 1348 RSB
```



```
042A 1350 .SBTTL CNCT_CHECK - Verify Connect Data
042A 1351 :++
042A 1352 : FUNCTIONAL DESCRIPTION:
042A 1353 :
042A 1354 : Evaluate received connect data vs. connect data
042A 1355 : sent or about to be sent.
042A 1356 :
042A 1357 : CALLING SEQUENCE:
042A 1358 :
042A 1359 : BSBW CNCT_CHECK
042A 1360 : IPL must be at IPL$SCS
042A 1361 :
042A 1362 : INPUT PARAMETERS:
042A 1363 :
042A 1364 : R2 Address of received connect data message
042A 1365 : R5 Address of CSB
042A 1366 :
042A 1367 : OUTPUT PARAMETERS:
042A 1368 :
042A 1369 : R0 is status
042A 1370 : TRUE implies all is well, ACCEPT or proceed with connection
042A 1371 : FALSE implies incompatibility, REJECT or break connection
042A 1372 : requesting remote node to BUGCHECK
042A 1373 :
042A 1374 : SIDE EFFECTS:
042A 1375 :
042A 1376 : This node will BUGCHECK if incompatible with the remote node and
042A 1377 : it appears "best" that this node exit.
042A 1378 :
042A 1379 : R1 is destroyed.
042A 1380 :--
042A 1381 :
042A 1382 : CNCT_CHECK:
1C BB 042A 1383 : PUSH R2,R3,R4 ; Save registers
042C 1384 :
042C 1385 : First, check message size of remote system against size required by
042C 1386 : clusters.
042C 1387 :
042C 1388 : MOVL CSB$L_CLUB(R3),R4 ; Address of CLUB
0430 1389 : MOVAB CSB$B_CNCT(R5),R3 ; Address of my connect data
0434 1390 : MOVAB SCSCMSG$B_SNDAT(R2),R2 ; Address of remote connect data
0438 1391 :
0438 1392 : CMPB CSB$B_VERNUM(R5), - ; Compare remote version number to
043D 1393 : CNCT$B_VERNUM(R2) ; local version number
043D 1394 : BGEQU 10$ ; Branch if remote is >= local
043F 1395 :
043F 1396 : Get here is local node has high version number than remote node
043F 1397 : If the versions are compatible, branch to 50$
043F 1398 : As of now, all different versions are incompatible.
043F 1399 : If the versions are incompatible, decide who should crash.
043F 1400 :
28 0A A2 00 E0 043F 1401 : BBS #CNCT$V_CLUSTER, - ; Branch if remote node is a
0444 1402 : CNCT$B_CLSSTS(R2),70$ ; cluster member
1C 11 0444 1403 : BRB 40$ ; Branch to failure exit
0446 1404 :
0446 1405 : Get here if version are identical or if remote is a newer (higher)
0446 1406 : protocol than local.
```

```
006B 8F 00000000'GF B1 0446 1407 ; Check compatibility of message buffer sizes.
                                0446 1408 ;
                                0446 1409 10$: CMPW G^SCSS$GW_MAXMSG, - ; Is local system's maximum message
                                044F 1410 #CLMSG$K_MAXMSG ; size big enough?
                                044F 1411 BLSSU 90$ ; Branch if too small
                                0451 1412 MOVL CSB$L_SB(R5),R0 ; SB address of remote system
006B 8F 50 68 A5 D0 0455 1413 CMPW S$W_MAXMSG(R0), - ; Compare against maximum cluster
                                045B 1414 #CLMSG$K_MAXMSG
                                045B 1415 BGEQU 50$ ; Branch if it is big enough
0E 0A A2 09 1E 045B 1415 BBS #CNCT$V_CLUSTER, - ; Branch if remote node is
                                045D 1416 CNCT$B_CLSSTS(R2),80$ ; a cluster member
                                0462 1417
                                0462 1418
                                0462 1419 40$: CLRL R0 ; Form failure status
                                0464 1420 BRB 60$ ; Branch to common exit
                                0466 1421
                                0466 1422 50$: MOVL S^#SS$_NORMAL,R0 ; Success status
00 00' D0 0466 1422 60$: POPR #^M<R2,R3,R4> ; Restore registers
1C 1C BA 0469 1423 RSB
05 05 046B 1424
                                046C 1425
                                046C 1426 ;
                                046C 1427 ; Get here when node must leave cluster
                                046C 1428 ;
                                046C 1429 70$: BUG_CHECK CLUEXIT,FATAL ; Leave cluster because of incompatible
                                0470 1430 ; protocol levels
                                0470 1431
                                0470 1432 80$: BUG_CHECK CLUEXIT,FATAL ; Remote node is a cluster member with an
                                0474 1433 ; insufficient message buffer size. This
                                0474 1434 ; node may never join, so die.
                                0474 1435
                                0474 1436 90$: BUG_CHECK CLUEXIT,FATAL ; Local node has too small value of SYSGEN
0478 1437 ; parameter SCSMAXMSG.
```

```
0478 1439 .SBTTL RECNECT_CHECK - Verify Reconnect Data
0478 1440 :++
0478 1441 : FUNCTIONAL DESCRIPTION:
0478 1442 :
0478 1443 : Evaluate received reconnect data vs. connect data
0478 1444 : sent or about to be sent.
0478 1445 :
0478 1446 : CALLING SEQUENCE:
0478 1447 :
0478 1448 : BSBW RECNECT_CHECK
0478 1449 : IPL must be at IPL$_SCS
0478 1450 :
0478 1451 : INPUT PARAMETERS:
0478 1452 :
0478 1453 : R5 Address of CSB
0478 1454 : R2 Address of SCS connect message
0478 1455 :
0478 1456 : OUTPUT PARAMETERS:
0478 1457 :
0478 1458 : None
0478 1459 :
0478 1460 : SIDE EFFECTS:
0478 1461 :
0478 1462 : R0 and R1 are destroyed.
0478 1463 : --
0478 1464 :
0478 1465 : Legend:
0478 1466 :
0478 1467 : C: CLUB CLUSTER bit: this node is cluster member
0478 1468 : c: CNCT CLUSTER bit: other node is cluster member
0478 1469 : Q: CLUB QUORUM bit: cluster containing this node has quorum
0478 1470 : q: CNCT QUORUM bit: cluster containing other node has quorum
0478 1471 : M: CSB MEMBER bit: connection is to local cluster member
0478 1472 : m: CNCT MEMBER bit: connection is to local cluster member
0478 1473 : L: CSB LONG BREAK bit: this node has seen long cnx break
0478 1474 : l: CNCT LONG BREAK bit: other node has seen long cnx break
0478 1475 : R: CSB REMOVED bit: this node has removed other from cluster
0478 1476 : r: CNCT REMOVED bit: other node has removed this node from cluster
0478 1477 :
0478 1478 RECNECT_CHECK:
0478 1479 : PUSH R2,R3 : Save registers
0478 1480 : MOVL CSB$_CLUB(R5),R3 : Get CLUB address
0478 1481 : MOVAB SCSCMSG$_SNDDAT(R2),R2 : Address of received connect data
0478 1482 :
0478 1483 : If other node has seen long break, make sure this node counts it as a
0478 1484 : long break also.
0478 1485 :
0478 1486 : BBC #CNCT$_V_LONG_BREAK, - : Branch if (NOT L) & (NOT l)
0478 1487 : CNCT$_B_CNXTS(R2),10$ :
0478 1488 : BSBB LONG_BREAK : Treat as though a long break
0478 1489 : BBS #CLUB$_V_CLUSTER, - : Branch if local node is cluster
0478 1490 : CLUB$_L_FLAGS(R3),15$ : member
0478 1491 : BBS #CSB$_V_LONG_BREAK, - : Branch if long break and bugcheck
0478 1492 : CSB$_L_STATUS(R5),80$ : (NOT C) & L
0478 1493 : BRB 50$ : All seems well
0478 1494 :
0478 1495 : BBC #CNCT$_V_REMOVED, - : Branch if other has not removed us
```

```
04 A2 06 A2 B1 049A 1496 CNCT$B_CNXSTS(R2),20$ ;
049A 1497 CMPW CNCT$W_VOTES(R2), - ; Does remote cluster have a quorum?
049F 1498 CNCT$W_QUORUM(R2)
07 60 A5 45 1E 049F 1499 70$ ; Branch if r & q and bugcheck
07 60 A5 02 E1 04A1 1500 20$: BBC #CSBSV_REMOVED, - ; Branch if we have not removed other
04A6 1501 CSBSL_STATUS(R5),30$
20 A3 22 A3 B1 04A6 1502 CMPW CLUB$W_VOTES(R3), - ; Does local cluster have a quorum?
04AB 1503 CLUB$W_QUORUM(R3)
2F 1E 04AB 1504 BGEQU 40$ ; Branch if we have quorum --
04AD 1505 ; other should go: R & Q
04AD 1506 30$:
04AD 1507
04AD 1508 : In the following two instructions, consider the case where
04AD 1509 one node has removed the other, but not vice-versa. In these cases,
04AD 1510 it seems necessary that outgoing messages to the other node be blocked.
04AD 1511 In actuality, LONG_BREAK should inhibit outgoing messages from all
04AD 1512 parties except the connection manager. The inhibited messages should
04AD 1513 be immediately returned with error status.
04AD 1514
2E 60 A5 02 E1 04AD 1515 BBC #CSBSV_REMOVED, - ; Branch if we have not removed
04B2 1516 CSBSL_STATUS(R5),50$ ; other node
29 0B A2 02 E1 04B2 1517 BBC #CNCT$V_REMOVED, - ; Branch if other node has not removed
04B7 1518 CNCT$B_CNXSTS(R2),50$ ; this node
04B7 1519
04B7 1520 : Each node has removed the other
04B7 1521 : Neither node has quorum: R & r & (NOT Q) & (NOT q)
04B7 1522
22 A3 06 A2 B1 04B7 1523 CMPW CNCT$W_VOTES(R2), - ; Compare available votes
04BC 1524 CLUB$W_VOTES(R3)
28 1A 04BC 1525 BGTRU 70$ ; Other has more votes, we crash
1C 1F 04BE 1526 BLSSU 40$ ; This node has more votes
24 A3 08 A2 B1 04C0 1527 CMPW CNCT$W_NODES(R2), - ; Compare number of nodes in cluster
04C5 1528 CLUB$W_NODES(R3)
1F 1A 04C5 1529 BGTRU 70$ ; Other has more nodes, crash
13 1F 04C7 1530 BLSSU 40$ ; Other nodes has more nodes
0C BB 04C9 1531 PUSHR #^M<R2,R3>
52 68 A5 D0 04CB 1532 MOVL CSBSL_SB(R5),R2 ; Remote System Block
00000018'GF 06 29 04CF 1533 CMPC3 #SB$S_SYSTEMID, - ; Compare system ID's
18 A2 04D6 1534 G^SCS$GA_LOCALSB+SB$B_SYSTEMID, - ; local system ID
04D8 1535 SB$B_SYSTEMID(R2) ; remote system ID
0C BA 04D8 1536 POPR #^M<R2,R3>
0A 1A 04DA 1537 BGTRU 70$
50 D4 04DC 1538 40$: CLRL R0 ; Failure status means other
03 11 04DE 1539 BRB 60$ ; node should bugcheck
04E0 1540
50 01 D0 04E0 1541 50$: MOVL #1,R0
OC BA 04E3 1542 60$: POPR #^M<R2,R3> ; Restore registers
05 05 04E5 1543 RSB
04E6 1544
04E6 1545 : Get here when node must leave cluster
04E6 1546
04E6 1547
04E6 1548 70$: BUG_CHECK CLUEXIT,FATAL ; Leave cluster
04EA 1549
04EA 1550
04EA 1551 : Get here when two nodes not part of a cluster regain a connection after
04EA 1552 : a long break.
```


M 10

- Cluster Connection Manager
RECNCT_CHECK - Verify Reconnect Data

Page 34
(17)

04EA 1553 :
04EA 1554 808: BUG_CHECK CLUEXIT,FATAL ; Reboot to avoid inconsistency

CN
Sy

CS
CS
CS
CS
CS
CS
CS
CS
CS
CS
CS
CS
CS
CS
CS
CS
CS
CS
CS
CS
CS
CS
CS
CS
CS
CS
DE
DE
DE
DI
DY
DY
DY
EX
EX
EX
EX
FA
FK
IP
IP
IP
LI
LO
PB
PO

```
04EE 1556 .SBTTL LONG_BREAK - Long Break in Connection
04EE 1557 ++
04EE 1558 FUNCTIONAL DESCRIPTION:
04EE 1559
04EE 1560 Connection has remained broken for a long time.
04EE 1561 All un-acked messages have their fork process resumed with
04EE 1562 a failure status.
04EE 1563
04EE 1564 CALLING SEQUENCE:
04EE 1565
04EE 1566 BSBW LONG_BREAK
04EE 1567 IPL must be at IPL$_SCS
04EE 1568
04EE 1569 INPUT PARAMETERS:
04EE 1570
04EE 1571 R5 Address of CSB
04EE 1572
04EE 1573 OUTPUT PARAMETERS:
04EE 1574
04EE 1575 None
04EE 1576
04EE 1577 SIDE EFFECTS:
04EE 1578
04EE 1579 R0 and R1 are destroyed.
04EE 1580 --
04EE 1581
04EE 1582 LONG_BREAK:
OB 60 A5 00 E2 04EE 1583 BBSS #CSB$_V LONG_BREAK, - ; Mark long break seen and
04F3 1584 CSB$_L STATUS(R5), 10$ ; return if already set
50 0000'CF 9E 04F3 1585 MOVAB FAILIO MSG,R0 ; Address of message
04F8 1586 BSBW CNX$CONFIGN CHANGE ; Note configuration change
04FB 1587 BSBW CNX$CON BREAK ; Report long break
07 43 A5 91 04FE 1588 10$: CMPB CSB$_B STATE(R5), - ; Are we disconnecting?
0502 1589 #CSB$_R_DISCONNECT
0502 1590 BEQL 20$ ; Branch if yes
FAF9' 30 0504 1591 BSBW CNX$FAIL_MSG ; Complete outstanding messages
05 0507 1592 20$: RSB
```

		0508 1594	.SBTTL CNX\$DECREFCNT - Decrement CSB Reference Count		
		0508 1595			
		0508 1596	:++		
		0508 1597	: FUNCTIONAL DESCRIPTION:		
		0508 1598			
		0508 1599	This routine decrements a CSB reference count and deletes		
		0508 1600	the CSB when the reference count goes to 0.		
		0508 1601			
		0508 1602	: CALLING SEQUENCE:		
		0508 1603			
		0508 1604	BSBB CNX\$DECREFCNT		
		0508 1605			
		0508 1606	: INPUT PARAMETERS:		
		0508 1607			
		0508 1608	R5 Address of CSB		
		0508 1609			
		0508 1610	: IMPLICIT INPUTS:		
		0508 1611			
		0508 1612	NONE		
		0508 1613			
		0508 1614	: OUTPUT PARAMETERS:		
		0508 1615	R5: Address of CSB, if not deleted		
		0508 1616	R5: Contents of CSB\$SYSQBL, if CSB deleted		
		0508 1617			
		0508 1618	: IMPLICIT OUTPUTS:		
		0508 1619	NONE		
		0508 1620			
		0508 1621	: COMPLETION CODES:		
		0508 1622	NONE		
		0508 1623			
		0508 1624	: SIDE EFFECTS:		
		0508 1625	R2-R4 preserved		
		0508 1626			
		0508 1627	:--		
		0508 1628			
		0508 1629	CNX\$DECREFCNT::		
6C A5	97	0508 1630	DECB	CSB\$B_REF_CNT(R5)	: Decrement reference count
2A	12	0508 1631	BNEQ	20\$: Branch if non-zero
FAF0'	30	050D 1632	BSBW	CNX\$FAIL MSG	: Fail any outstanding messages
26	10	0510 1633	BSBB	DELETE TQE	: Flush timer queue entry
50 0000'CF	9E	0512 1634	MOVAB	DEAD MSG,R0	: Address of dead node message
FAE6'	30	0517 1635	BSBW	CNX\$CONFIG CHANGE	: Report configuration change
50 68 A5	D0	051A 1636	MOVL	CSB\$SB(R5),R0	: SB address
5C A0 55	D1	051E 1637	CMPL	R5,SB\$CSB(R0)	: Is this CSB pointed to?
03	12	0522 1638	BNEQ	10\$	
5C A0	D4	0524 1639	CLRL	SB\$CSB(R0)	: Invalidate back pointer
04 A5	DD	0527 1640	PUSHL	CSB\$SYSQBL(R5)	: Backward link
0C	BB	052A 1641	PUSHR	#^M<R2,R3>	: Save registers
50 65	OF	052C 1642	REMQUE	(R5),R0	: Unlink CSB
00000000'GF	16	052F 1643	JSB	G^EX\$DEANONPAGED	: Deallocate it
2C	BA	0535 1644	POPR	#^M<R2,R3,R5>	: Restore registers
	05	0537 1645	RSB		

```
0538 1647 .SBTTL DELETE_TQE - Delete a TQE linked to a CSB
0538 1648
0538 1649 :++
0538 1650 : FUNCTIONAL DESCRIPTION:
0538 1651 :
0538 1652 : This routine deallocates a TQE linked to a CSB.
0538 1653 :
0538 1654 : CALLING SEQUENCE:
0538 1655 :
0538 1656 : BSBB DELETE_TQE
0538 1657 :
0538 1658 : INPUT PARAMETERS:
0538 1659 :
0538 1660 : R5 Address of CSB
0538 1661 :
0538 1662 : IMPLICIT INPUTS:
0538 1663 :
0538 1664 : CSB$L_TQE is 0 or address of TQE
0538 1665 :
0538 1666 : OUTPUT PARAMETERS:
0538 1667 : NONE
0538 1668 :
0538 1669 : IMPLICIT OUTPUTS:
0538 1670 : NONE
0538 1671 :
0538 1672 : COMPLETION CODES:
0538 1673 : NONE
0538 1674 :
0538 1675 : SIDE EFFECTS:
0538 1676 : R0 and R1 destroyed.
0538 1677 :
0538 1678 :--
0538 1679
0538 1680 DELETE_TQE:
50 44 A5 D0 0538 1681 MOVL CSB$L_TQE(R5),R0 : Get address of TQE
10 13 0538 1682 BEQL 10$ : There isn't one
OC BB 0538 1683 PUSHF #^M<R2,R3> : Save registers
50 60 OF 0540 1684 REMQUE (R0),R0 : Remove from timer queue
00000000 GF 16 0543 1685 JSB G^EXE$DEANONPAGED : Deallocate it
44 A5 C4 0549 1686 CLRL CSB$L_TQE(R5) : Clear pointer
OC BA 054C 1687 POPF #^M<R2,R3> : Restore registers
05 054E 1688 10$: RSB
```



```
054F 1690 .SBTTL DEAD_NODE - Manage death of a node
054F 1691 :++
054F 1692 : FUNCTIONAL DESCRIPTION:
054F 1693 :
054F 1694 :     Called when a new incarnation of a system has been seen to
054F 1695 :     remove the last vestiges of knowledge of the old incarnation.
054F 1696 :     No connection to the system exists. The CSB is deleted.
054F 1697 :
054F 1698 : CALLING SEQUENCE:
054F 1699 :
054F 1700 :     BSBB    DEAD_NODE
054F 1701 :
054F 1702 : INPUT PARAMETERS:
054F 1703 :
054F 1704 :     R5      Address of CSB
054F 1705 :
054F 1706 : IMPLICIT INPUTS:
054F 1707 :
054F 1708 :     NONE
054F 1709 :
054F 1710 : OUTPUT PARAMETERS:
054F 1711 :
054F 1712 :     NONE
054F 1713 :
054F 1714 : IMPLICIT OUTPUTS:
054F 1715 :     NONE
054F 1716 :
054F 1717 : COMPLETION CODES:
054F 1718 :     NONE
054F 1719 :
054F 1720 : SIDE EFFECTS:
054F 1721 :     R0 and R1 are destroyed. R5 is invalidated.
054F 1722 :
054F 1723 : --
054F 1724 :
054F 1725 : DEAD_NODE:
9D 10 054F 1726 :     BSBB    LONG BREAK          : Simulate connection broken for long time
E5 10 0551 1727 :     BSBB    DELETE_TQE          : Get rid of timer
B3 10 0553 1728 :     BSBB    CNX$DECREFCNT       : Get rid of CSB (eventually)
05 05 0555 1729 :     RSB
```

```
0556 1731 .SBTTL CNX_STATUS_CHECK - Check SCS failure message
0556 1732 ++
0556 1733 FUNCTIONAL DESCRIPTION:
0556 1734
0556 1735 Check SCS failure message and BUGCHECK if the remote node has requested
0556 1736 it.
0556 1737
0556 1738 CALLING SEQUENCE:
0556 1739
0556 1740 JSB CNX_STATUS_CHECK
0556 1741
0556 1742 INPUT PARAMETERS:
0556 1743
0556 1744 R5 CSB address
0556 1745 R0 SCS Reason code
0556 1746 R1 SYSAP reason (if R0=SS$_REJECT)
0556 1747
0556 1748 IMPLICIT INPUTS:
0556 1749
0556 1750 NONE
0556 1751
0556 1752 OUTPUT PARAMETERS:
0556 1753
0556 1754 NONE
0556 1755
0556 1756 IMPLICIT OUTPUTS:
0556 1757
0556 1758 NONE
0556 1759
0556 1760 COMPLETION CODES:
0556 1761
0556 1762 NONE
0556 1763
0556 1764 SIDE EFFECTS:
0556 1765
0556 1766 NONE
0556 1767
0556 1768 --
0556 1769
0556 1770 CNX_STATUS_CHECK:
0556 1771
0556 1772 PUSH R0,R1 ; Save registers
0556 1773 CMP R0,SS$_REJECT ; Is this a connection request reject?
0556 1774 BEQ 10$ ; Branch if yes
0556 1775 CMP R0,SS$_DISCONNECT ; Is this a requested disconnect?
0556 1776 BNEQ 20$ ; Branch if no
0556 1777 BBC CLMDRSSV_DRS,R1,20$ ; Branch if not a cluster disconnect code
0556 1778 BBS CLMDRSSV_FATAL,R1,50$ ; Branch if bugcheck requested
0556 1779 CMP R0,SS$_DISCONNECT ; Is this a requested disconnect?
0556 1780 BNEQ 20$ ; Branch if no
0556 1781 CMPB CLMDRSSC_REMOVED,R1 ; Is this node removed from cluster?
0556 1782 BEQ 40$ ; Branch if local node removed and exit
0556 1783 CMP R0,SS$_VCBROKEN ; Is this a circuit failure?
0556 1784 BNEQ 30$ ; Branch if no
0556 1785 CMP R1,SS$_NOSUCHNODE ; Is this a result of a 'last gasp'?
0556 1786 BNEQ 30$ ; Branch if no
0556 1787 BSBW LONG_BREAK ; Declare a long break
0556 1788 POP R0,R1 ; Restore registers
0556 1789
```

0000'8F	03	BB	0556	1771	PUSH	R0,R1	;	Save registers
	50	B1	0558	1772	CMPI	R0,SS\$_REJECT	;	Is this a connection request reject?
	07	13	055D	1773	BEQ	10\$;	Branch if yes
0000'8F	50	B1	055F	1774	CMPI	R0,SS\$_DISCONNECT	;	Is this a requested disconnect?
	14	12	0564	1775	BNEQ	20\$;	Branch if no
10 51	0F	E1	0566	1776	BBC	CLMDRSSV_DRS,R1,20\$;	Branch if not a cluster disconnect code
24 51	08	E0	056A	1777	BBS	CLMDRSSV_FATAL,R1,50\$;	Branch if bugcheck requested
0000'8F	50	B1	056E	1778	CMPI	R0,SS\$_DISCONNECT	;	Is this a requested disconnect?
	05	12	0573	1779	BNEQ	20\$;	Branch if no
51	0A	91	0575	1780	CMPI	CLMDRSSC_REMOVED,R1	;	Is this node removed from cluster?
	14	13	0578	1781	BEQ	40\$;	Branch if local node removed and exit
0000'8F	50	B1	057A	1782	CMPI	R0,SS\$_VCBROKEN	;	Is this a circuit failure?
	0A	12	057F	1783	BNEQ	30\$;	Branch if no
0000'8F	51	B1	0581	1784	CMPI	R1,SS\$_NOSUCHNODE	;	Is this a result of a 'last gasp'?
	03	12	0586	1785	BNEQ	30\$;	Branch if no
FF63	30		0588	1786	BSBW	LONG_BREAK	;	Declare a long break
03	BA		058B	1787	POP	R0,R1	;	Restore registers

CNXMAN
V04-000

F 11

- Cluster Connection Manager		16-SEP-1984 00:24:50	VAX/VMS Macro V04-00	Page 40
CNX_STATUS_CHECK - Check SCS failure mes		5-SEP-1984 04:07:15	[SYSLOA.SRC]CNXMAN.MAR;1	(22)

05	058D	1788	RSB		; Return to caller
	058E	1789			
	058E	1790	40\$:	BUG_CHECK	CLUEXIT,FATAL ; This node removed from cluster
	0592	1791			
	0592	1792	50\$:	BUG_CHECK	CNXMGRERR,FATAL ; Bugcheck requested by disconnecting remote
	0596	1793			

```
0596 1795 .SBTTL CNX$LOOKUP_CSB - Lookup a CSB given a SB address
0596 1796 ++
0596 1797 FUNCTIONAL DESCRIPTION:
0596 1798
0596 1799 CNX$LOOKUP_CSB find a CSB with matching System ID and
0596 1800 software incarnation number given an SB address.
0596 1801
0596 1802 CALLING SEQUENCE:
0596 1803
0596 1804 JSB CNX$LOOKUP_CSB
0596 1805
0596 1806 INPUT PARAMETERS:
0596 1807
0596 1808 R1 Address of SB
0596 1809
0596 1810 IMPLICIT INPUTS:
0596 1811
0596 1812 NONE
0596 1813
0596 1814 OUTPUT PARAMETERS:
0596 1815
0596 1816 R5 is address of CSB
0596 1817
0596 1818 IMPLICIT OUTPUTS:
0596 1819
0596 1820 NONE
0596 1821
0596 1822 COMPLETION CODES:
0596 1823
0596 1824 R0 contains status
0596 1825
0596 1826 SIDE EFFECTS:
0596 1827
0596 1828 R1 is destroyed
0596 1829
0596 1830 --
0596 1831
0596 1832 CNX$LOOKUP_CSB::
0596 1833
0596 1834 PUSH #M<R2,R3,R4> ; Lookup given SB address
0596 1835 MOV R1,R4 ; Save registers
0596 1836 MOV SB$L_CSB(R4),R5 ; SB address
0596 1837 BEQL 30$ ; Get CSB for this SB
0596 1838 ; Branch if no CSB for this SB
0596 1839
0596 1840 ; Check software incarnation.
0596 1841
0596 1842 BBS #CSB$V_LOCAL, - ; Skip if local system
0596 1843 CSB$L_STATUS(R5),50$
0596 1844 CMPC3 #CSB$S_SWINCARN, - ; Software incarnations match?
0596 1845 CSB$Q_SWINCARN(R5),-
0596 1846 SB$Q_SWINCARN(R4)
0596 1847 BEQL 50$ ; Branch if yes and exit
0596 1848
0596 1849 ; There is an existing CSB with a different software incarnation.
0596 1850 ; Get rid of it and fail over that node (perhaps for the second time!)
0596 1851
0596 1852 STATE_DISP <<NEW,40$>,<DEAD,30$>,<WAIT,20$>,<RECONNECT,10$>>
0596 1853 BUG_CHECK CNXMGRERR,FATAL ; Temporary Bugcheck
0596 1854
0596 1855
0596 1856
0596 1857
0596 1858
0596 1859
0596 1860
0596 1861
0596 1862
0596 1863
0596 1864
0596 1865
0596 1866
0596 1867
0596 1868
0596 1869
0596 1870
0596 1871
0596 1872
0596 1873
0596 1874
0596 1875
0596 1876
0596 1877
0596 1878
0596 1879
0596 1880
0596 1881
0596 1882
0596 1883
0596 1884
0596 1885
0596 1886
0596 1887
0596 1888
0596 1889
0596 1890
0596 1891
0596 1892
0596 1893
0596 1894
0596 1895
0596 1896
0596 1897
0596 1898
0596 1899
0596 1900
0596 1901
0596 1902
0596 1903
0596 1904
0596 1905
0596 1906
0596 1907
0596 1908
0596 1909
0596 1910
0596 1911
0596 1912
0596 1913
0596 1914
0596 1915
0596 1916
0596 1917
0596 1918
0596 1919
0596 1920
0596 1921
0596 1922
0596 1923
0596 1924
0596 1925
0596 1926
0596 1927
0596 1928
0596 1929
0596 1930
0596 1931
0596 1932
0596 1933
0596 1934
0596 1935
0596 1936
0596 1937
0596 1938
0596 1939
0596 1940
0596 1941
0596 1942
0596 1943
0596 1944
0596 1945
0596 1946
0596 1947
0596 1948
0596 1949
0596 1950
0596 1951
0596 1952
0596 1953
0596 1954
0596 1955
0596 1956
0596 1957
0596 1958
0596 1959
0596 1960
0596 1961
0596 1962
0596 1963
0596 1964
0596 1965
0596 1966
0596 1967
0596 1968
0596 1969
0596 1970
0596 1971
0596 1972
0596 1973
0596 1974
0596 1975
0596 1976
0596 1977
0596 1978
0596 1979
0596 1980
0596 1981
0596 1982
0596 1983
0596 1984
0596 1985
0596 1986
0596 1987
0596 1988
0596 1989
0596 1990
0596 1991
0596 1992
0596 1993
0596 1994
0596 1995
0596 1996
0596 1997
0596 1998
0596 1999
0596 2000
```



```
06 11 05C6 1852 BRB 30% ; Branch to allocate new block
05C8 1853
43 0A 90 05C8 1854 20$: MOVB #CSB$K_DEAD,- ; Set state=DEAD
A5 10 05CA 1855 CSB$B_STATE(R5)
81 10 05CC 1856 BSBB DEAD_R0DE ; Handle dead node
51 54 D0 05CE 1857 30$: MOVL R4,RT ; SB address
OD 10 05D1 1858 BSBB CNX$CREATE_CSB ; Create new CSB
08 11 05D3 1859 BRB 60% ; Return with status
05D5 1860
38 A5 2C A4 7D 05D5 1861 40$: MOVQ SB$Q_SWINCARN(R4), - ; Update software incarnation and
05DA 1862 CSB$Q_SWINCARN(R5) ; continue
50 00' 3C 05DA 1863 50$: MOVZWL S^#SS$_NORMAL,R0 ; Found CSB, in R5
1C BA 05D3 1864 60$: POPR #^M<R2,R3,R4> ; Restore nonvolatile registers
05 05DF 1865 RSB
```

```
05E0 1867 .SBTTL CNX$CREATE_CSB - Create a new CSB given a SB address
05E0 1868
05E0 1869 ++
05E0 1870 FUNCTIONAL DESCRIPTION:
05E0 1871 CNX$CREATE_CSB creates a CSB with matching System ID and
05E0 1872 software incarnation number given an SB address.
05E0 1873 It is assumed that no similar CSB already exists.
05E0 1874
05E0 1875 CALLING SEQUENCE:
05E0 1876 JSB CNX$CREATE_CSB
05E0 1877
05E0 1878 INPUT PARAMETERS:
05E0 1879 R1 Address of SB
05E0 1880
05E0 1881 IMPLICIT INPUTS:
05E0 1882 NONE
05E0 1883
05E0 1884 OUTPUT PARAMETERS:
05E0 1885 R5 is address of CSB
05E0 1886
05E0 1887 IMPLICIT OUTPUTS:
05E0 1888 NONE
05E0 1889
05E0 1890 COMPLETION CODES:
05E0 1891 R0 contains status
05E0 1892
05E0 1893 SIDE EFFECTS:
05E0 1894 R1 is destroyed
05E0 1895
05E0 1896
05E0 1897
05E0 1898
05E0 1899
05E0 1900 --
05E0 1901
05E0 1902 CNX$CREATE_CSB::
51 00DC 8F BB 05E0 1903 PUSH R2,R3,R4,R6,R7 ; Lookup given SB address
57 51 DO 05E4 1904 MOVL R1,R7 ; Save registers
00AC 8F 3C 05E7 1905 MOVZWL #CSB$K_LENGTH,R1 ; SB address
FA11' 30 05EC 1906 BSBW CNX$ALCOZMEM ; Size of CSB
03 50 E8 05EF 1907 BLBS R0,10$ ; Allocate and zero memory
009C 31 05F2 1908 BRW 50$ ; Branch if successful
; Exit, status in R0
56 52 DO 05F5 1909 10$: MOVL R2,R6 ; CSB address
5C A7 56 DO 05F8 1910 MOVL R6,SB$C_CSB(R7) ; Update SB to point to newest CSB
01 90 05FC 1911 MOV B #DYN$C_CLU_CSB,- ; Store subtype
0B A6 05FE 1912 CSB$B_SUBTYPE(R6)
14 A6 DE 0600 1913 MOVAL CSB$C_SENTQFL(R6),- ; Initialize sent list
18 A6 DE 0603 1914 CSB$C_SENTQBL(R6)
14 A6 D4 0605 1915 CLRL CSB$C_SENTQFL(R6)
1C A6 DE 0608 1916 MOVAL CSB$C_RESENDQFL(R6),- ; Initialize resend list
20 A6 DE 060B 1917 CSB$C_RESENDQBL(R6)
1C A6 D4 060D 1918 CLRL CSB$C_RESENDQFL(R6)
34 A6 01 DO 0610 1919 MOVL #1,CSB$C_CURRCDRP(R6) ; Block critical section in SEND_MSG
24 A6 9E 0614 1920 MOVAB CSB$C_WARMCDRPQFL(R6),- ; Initialize warm CDRP queue
24 A6 0617 1921 CSB$C_WARMCDRPQFL(R6)
24 A6 9E 0619 1922 MOVAB CSB$C_WARMCDRPQFL(R6),-
```

```
28 A6      061C 1924      CSBSL_WARMCDRPQBL(R6)
           061E 1925
           061E 1926      ; Store remote side's software incarnation number and system id.
           061E 1927      ; so that if this connection breaks and another is established,
           061E 1928      ; we can determine if it's the same system and software
           061E 1929      ; incarnation at the other end.
           061E 1930
           2C A7 7D 061E 1931      MOVQ  SBSQ SWINCARN(R7), -      ; Store software incarnation number
           38 A6 90 0621 1932      CSBSQ SWINCARN(R6)
           04 90 0623 1933      MOVB  #CSBSR_NEW, -      ; Set state to NEW
           43 A6 90 0625 1934      CSBSB_STATE(R6)
           58 A6 DE 0627 1935      MOVAL CSBSL_PARTNERQFL(R6), -      ; Initialize block transfer
           SC A6 58 A6 DE 062C 1936      CSBSL_PARTNERQFL(R6), -      ; partners queue.
           DE 062C 1937      MOVAL  CSBSL_PARTNERQFL(R6), -
           74 A6 00000000'GF 7D 0631 1938      CSBSL_PARTNERQBL(R6)
           64 A6 00000000'GF D0 0639 1939      MOVQ  G^EXESQ SYS TIME, -      ; Stamp reference time in CSB
           6C A6 01 90 0639 1940      CSBSQ REFTIME(R6)
           68 A6 57 D0 0641 1941      MOVL  G^CLUSGL CLUB, -      ; Address of CLUB
           50 7C A6 9E 0641 1942      CSBSL_CLOB(R6)
           60 00'8F 90 0641 1943      MOVB  #1, CSBSB_REF CNT(R6)      ; Initialize reference count
           01 A0 0C 90 0645 1944      MOVL  R7, CSBSL_SB(R6)      ; Address of SB
           02 A0 01 90 0649 1945      MOVAB CSBSB_CNCT(R6), R0      ; Connect data block
           03 A0 04 90 064D 1946      MOVB  I^#0, CNCT$B_EC0LV(LR0)      ; ECO level, set for easy patching
           55 64 A6 D0 0651 1947      MOVB  #CNCT$K_PROTOCOL, -      ; Protocol level
           64 A6 55 D0 0655 1948      CNCT$B_VERNUM(R0)
           50 68 A5 D0 0655 1949      MOVB  #CNCT$K_INITIAL, -      ; Initial connect
           18 A0 18 A7 06 29 0659 1950      CNCT$B_TYPE(R0)
           04 B5 66 0E 0659 1951      MOVB  #SEND_CREDITS-1, -      ; Unacknowledged message limit is
           55 56 D0 065D 1952      CNCT$B_ACKLIM(R0)      ; send credits - 1.
           50 0000'CF 9E 065D 1953      ASSUME CLUB$CLUBQFL EQ 0
           F972' 30 065D 1954      ASSUME CSBSL_SYSQFL EQ 0
           50 00DC 8F BA 0661 1955      MOVL  CSBSL_CLUB(R6), R5      ; Get address of CSB queue header
           04 B5 66 0E 0661 1956      MOVL  CSBSL_SYSQFL(R5), R5      ; Get address of next CSB
           55 56 D0 0664 1957      CMPL  R5, CSBSL_CLUB(R6)      ; Reached end of list?
           50 00DC 8F BA 0668 1958      BEQL  30$      ; Yes
           18 A0 18 A7 06 29 066A 1959      MOVL  CSBSL_SB(R5), R0      ; This CSB's SB address
           04 B5 66 0E 066E 1960      CMPC3 #SB$S_SYSTEMID, -      ; Compare system IDs
           55 56 D0 0674 1961      SB$B_SYSTEMID(R7), -
           50 00DC 8F BA 0674 1962      SB$B_SYSTEMID(R0)
           04 B5 66 0E 0674 1963      BGTR  20$      ; Branch if no match
           55 56 D0 0676 1964      INSQUE CSBSL_SYSQFL(R6), -
           50 00DC 8F BA 067A 1965      @CSBSL_SYSQBL(R5)
           04 B5 66 0E 067A 1966      MOVL  R6, R5      ; Set up result register
           55 56 D0 067D 1967      CMPL  R7, #SCS$GA_LOCALSB      ; Is this the local SB?
           50 00DC 8F BA 0684 1968      BEQL  40$      ; Skip message output
           04 B5 66 0E 0686 1969      MOVAB CSB_MSG, R0      ; because we are at IPL 31!
           55 56 D0 0686 1970      BSBW  CNX$CONFIG CHANGE      ; Address of new CSB message
           50 00DC 8F BA 068B 1971      MOVZWL S^#SS$_NORMAL, R0      ; Log CSB creation
           04 B5 66 0E 068E 1972      POPR  #M<R2,R3,R4,R6,R7>      ; Found CSB, in R5
           55 56 D0 0691 1973      RSB      ; Restore nonvolatile registers
           50 00DC 8F BA 0695 1974
```

```
0696 1976 .SBTTL DISPATCH - Dispatch on CSB state
0696 1977 :++
0696 1978 :FUNCTIONAL DESCRIPTION:
0696 1979 :
0696 1980 :   This routine dispatches based upon the CSB state
0696 1981 :   using a table assembled in-line at the call site.
0696 1982 :   The STATE_DISP macro defines and builds the table.
0696 1983 :
0696 1984 :INPUT PARAMETERS:
0696 1985 :
0696 1986 :   (SP)   First byte of dispatch table
0696 1987 :   R5     Address of CSB
0696 1988 :
0696 1989 :OUTPUT PARAMETERS:
0696 1990 :
0696 1991 :   NONE
0696 1992 :
0696 1993 :SIDE EFFECTS:
0696 1994 :
0696 1995 :   All registers preserved.
0696 1996 :--
0696 1997 :
0696 1998 DISPATCH::
0696 1999 PUSHR    #^M<R0,R1>           ; Save R0 and R1
0696 2000 MOVL     8(SP),R0            ; Fetch table address
0696 2001 10$: MOVZBL  (R0)+,R1     ; Fetch state code
0696 2002 BEQL     30$             ; End of table
0696 2003 CMPB     R1,CSB$B_STATE(R5) ; Match?
0696 2004 BEQL     20$             ; Branch on match
0696 2005 ADDL2    #2,R0           ; Bump over word offset
0696 2006 BRB      10$
0696 2007 06AC 2007
0696 2008 20$: CVTWL    (R0)+,R1     ; Fetch word offset
0696 2009 30$: ADDL3    R0,R1,8(SP) ; Store return address
0696 2010 POPR     #^M<R0,R1>       ; Restore registers
0696 2011 RSB
0696 2012
0696 2013
0696 2014 .END
```

50	08	03	BB	0696	1999				
	51	80	9A	0698	2000				
		0E	13	069C	2001	10\$:			
43	A5	51	91	069F	2002				
		05	13	06A1	2003				
	50	02	C0	06A5	2004				
		F0	11	06A7	2005				
				06AA	2006				
				06AC	2007				
08	AE	51	80	06AC	2008	20\$:			
		51	50	C1	06AF	30\$:			
		03	BA	06B4	2009				
			05	06B6	2010				
				06B7	2011				
				06B7	2012				
				06B7	2013				
				06B7	2014				

CNXMAN
Symbol table

- Cluster Connection Manager

L 11

16-SEP-1984 00:24:50 VAX/VMS Macro V04-00
5-SEP-1984 04:07:15 [SYSLOA.SRC]CNXMAN.MAR;1

Page 46
(25)

ACCPY MSG	*****	X	04	CNCT\$W_NODES	=	00000008		
BUG\$ CLUEXIT	*****	X	04	CNCT\$W_QUORUM	=	00000004		
BUG\$ CNXMGRERR	*****	X	04	CNCT\$W_RCVDSQNM	=	0000000C		
CDT\$C_AUXSTRUC	=	0000005C		CNCT\$W_VOTES	=	0C000006		
CDT\$C_PB	=	0000001C		CNCT_CHECK	0000042A	R	04	
CJFSMTH_JOURNAL	*****	X	03	CNCT_DATA	000003FF	R	04	
CLMDRSSC_PROTOCOL	=	00000004		CNCT_MSG	*****	X	04	
CLMDRSSC_REMOVED	=	0000000A		CNX\$ACCEPT	00000222	RG	04	
CLMDRSSC_VERSION	=	00000006		CNX\$ALLOZMEM	*****	X	03	
CLMDRSSM_DRS	=	00008000		CNX\$BREAK	00000309	RG	04	
CLMDRSSM_FATAL	=	00000100		CNX\$CHECK_QUORUM	*****	X	04	
CLMDRSSV_DRS	=	0000000F		CNX\$CONFIG_CHANGE	*****	X	04	
CLMDRSSV_FATAL	=	00000008		CNX\$CONNECT	00000038	RG	04	
CLSMMSG\$K_MAXMSG	=	0000006B		CNX\$CON_BREAK	*****	X	04	
CLUS\$GL_CLUB	*****	X	03	CNX\$CON_INIT	*****	X	03	
CLUS\$GL_CLUSVEC	*****	X	03	CNX\$CON_NEWSYS	*****	X	04	
CLUS\$GW_LCKDIRWT	*****	X	03	CNX\$CREATE_CSB	000005E0	RG	04	
CLUS\$GW_MAXINDEX	*****	X	03	CNX\$DECREFCNT	00000508	RG	04	
CLUS\$GW_QDSKVOTES	*****	X	03	CNX\$DISCONNECT	00000331	RG	04	
CLUS\$GW_QUORUM	*****	X	03	CNX\$DISC_BUGCHECK	000002CF	RG	04	
CLUS\$GW_RECINXINT	*****	X	04	CNX\$DISC_PROTOCOL	000002E1	RG	04	
CLUS\$GW_VOTES	*****	X	03	CNX\$DISC_REMOVE	000002D6	RG	04	
CLUB\$B_CLUBPWF	=	0000018C		CNX\$ERROR	000002F6	RG	04	
CLUB\$B_FORK_BLOCK	=	000000CC		CNX\$FAIL_MSG	*****	X	04	
CLUB\$B_SUBTYPE	=	0000000B		CNX\$INIT	00000000	RG	03	
CLUB\$K_LENGTH	=	000001A8		CNX\$LOOKUP_CSB	00000596	RG	04	
CLUB\$K_CSBQBL	=	00000004		CNX\$NEWSYSTEM	00000007	RG	04	
CLUB\$K_CSBQFL	=	00000000		CNX\$NEW_CSB	00000031	RG	04	
CLUB\$K_FLAGS	=	0000001C		CNX\$POST_CLEANUP	*****	X	04	
CLUB\$K_JNL_DISPT	=	00000014		CNX\$PRE_CLEANUP	*****	X	04	
CLUB\$K_LOCAL_CSB	=	00000010		CNX\$RCV_CNCT_MSG	0000015B	R	04	
CLUB\$K_POLL_CTX	=	0000000C		CNX\$RCV_MSG	*****	X	04	
CLUB\$M_CLUSTER	=	00000001		CNX\$RESEND_MSGS	*****	X	04	
CLUB\$V_CLUSTER	=	00000000		CNX\$WAIT	0000036A	RG	04	
CLUB\$W_FIRST_INDEX	=	00000066		CNX\$ERROR MSG	*****	X	04	
CLUB\$W_NODES	=	00000024		CNX STATUS_CHECK	00000556	R	04	
CLUB\$W_QDVOTES	=	000000AE		CSB\$B_CNCT	=	0000007C		
CLUB\$W_QUORUM	=	00000020		CSB\$B_ECOLVL	=	00000040		
CLUB\$W_VOTES	=	00000022		CSB\$B_REF_CNT	=	0000006C		
CLUB\$K\$B\$B_FORK_BLOCK	=	00000018		CSB\$B_REMACKLIM	=	00000033		
CLUBPWF\$B\$B_FORK_BLOCK	=	00000018		CSB\$B_STATE	=	00000043		
CNCT\$B_ACKLIM	=	00000003		CSB\$B_SUBTYPE	=	0000000B		
CNCT\$B_CLSSTS	=	0000000A		CSB\$B_VERNUM	=	00000041		
CNCT\$B_CNXTS	=	0000000B		CSB\$K_ACCEPT	=	00000006		
CNCT\$B_ECOLVL	=	00000000		CSB\$K_CONNECT	=	00000005		
CNCT\$B_TYPE	=	00000002		CSB\$K_DEAD	=	0000000A		
CNCT\$B_VERNUM	=	00000001		CSB\$K_DISCONNECT	=	00000007		
CNCT\$K_INITIAL	=	00000001		CSB\$K_LENGTH	=	000000AC		
CNCT\$K_PROTOCOL	=	0000000C		CSB\$K_LOCAL	=	0000000B		
CNCT\$K_RECONNECT	=	00000002		CSB\$K_NEW	=	00000004		
CNCT\$M_CLUSTER	=	00000001		CSB\$K_OPEN	=	00000001		
CNCT\$M_LONG_BREAK	=	00000001		CSB\$K_REACCEPT	=	00000008		
CNCT\$M_MEMBER	=	00000002		CSB\$K_RECONNECT	=	00000003		
CNCT\$M_REMOVED	=	00000004		CSB\$K_WAIT	=	00000009		
CNCT\$V_CLUSTER	=	00000000		CSB\$K_CDT	=	0000000C		
CNCT\$V_LONG_BREAK	=	00000000		CSB\$K_CLUB	=	00000064		
CNCT\$V_REMOVED	=	00000002		CSB\$K_CURRCORP	=	00000034		

CNXMAN
Symbol table

- Cluster Connection Manager

M 11

16-SEP-1984 00:24:50 VAX/VMS Macro V04-00
5-SEP-1984 04:07:15 [SYSLOA.SRC]CNXMAN.MAR;1

Page 47
(25)

CSBSL_PARTNERQBL	= 0000005C		
CSBSL_PARTNERQFL	= 00000058		
CSBSL_PDT	= 00000010		
CSBSL_RESENDQBL	= 00000020		
CSBSL_RESENDQFL	= 0000001C		
CSBSL_SB	= 00000068		
CSBSL_SENTQBL	= 00000018		
CSBSL_SENTQFL	= 00000014		
CSBSL_STATUS	= 00000060		
CSBSL_SYSQBL	= 00000004		
CSBSL_SYSQFL	= 00000000		
CSBSL_TIMEOUT	= 00000048		
CSBSL_TQE	= 00000044		
CSBSL_WARMCDRPQBL	= 00000028		
CSBSL_WARMCDRPQFL	= 00000024		
CSBSM_LOCAL	= 01000000		
CSBSM_LONG_BREAK	= 00000001		
CSBSM_MEMBER	= 00000002		
CSBSM_REMOVED	= 00000004		
CSBSQ_REFTIME	= 00000074		
CSBSQ_SWINCARN	= 00000038		
CSBSS_SWINCARN	= 00000008		
CSBSV_LOCAL	= 00000018		
CSBSV_LONG_BREAK	= 00000000		
CSBSV_REMOVED	= 00000002		
CSBSW_ACKRSEQNM	= 00000030		
CSBSW_LCKDIRWT	= 00000054		
CSBSW_QDVOTES	= 00000056		
CSBSW_QUORUM	= 00000052		
CSBSW_RCDVSEQNM	= 0000002E		
CSBSW_VOTES	= 00000050		
CSB_MSG	*****	X	04
DEAD_MSG	*****	X	04
DEAD_NODE	0000054F	R	04
DELETE_TQE	00000538	R	04
DISC_STATUS	000002E6	R	04
DISPATCH	00000696	RG	04
DYN\$C_CLU_CLUB	= 00000003		
DYN\$C_CLU_CLUVEC	= 00000002		
DYN\$C_CLU_CSB	= 00000001		
DYN\$C_TQE	= 0000000F		
EXESALONONPAGED	*****	X	04
EXESAL_TQENOREPT	*****	X	04
EXESDEANONPAGED	*****	X	04
EXESGL_ABSTIM	*****	X	04
EXESGO_SYSTIME	*****	X	04
EXESINSTIMQ	*****	X	04
FAILIO_MSG	*****	X	04
FKBSB_FIPL	= 0000000B		
FKBSK_LENGTH	= 00000018		
IPL\$SCS	= 00000008		
IPL\$SYNCH	= 00000008		
IPL\$TIMER	= 00000008		
LISTEN_ERROR	00000000	R	04
LONG_BREAK	000004EE	R	04
PBSL_SBLINK	= 00000030		
PDT\$C_REJECT	= 0000004C		

PROC_NAME	00000000	R	02
REACPT_MSG	*****	X	04
RECNET_CHECK	00000478	R	04
RECNET_MSG	*****	X	04
RETRY_CONNECT	000003DA	R	04
SBSB_SYSTEMID	= 00000018		
SBSL_CSB	= 0000005C		
SBSQ_SWINCARN	= 0000002C		
SBS\$SYSTEMID	= 00000006		
SBSW_MAXMSG	= 00000022		
SCSSACCEPT	*****	GX	04
SCSSCONFIG_SYS	*****	X	04
SCSSCONNECT	*****	X	04
SCSSDISCONNECT	*****	X	04
SCSSGA_LOCALSB	*****	X	03
SCSSGW_MAXMSG	*****	X	04
SCSSLISTEN	*****	GX	03
SCSSPOLL_MODE	*****	X	03
SCSSPOLL_PROC	*****	X	03
SCSCMSG\$B_SNDAT	= 00000020		
SEND_CREDITS	= 00000005		
SS\$DISCONNECT	*****	X	04
SS\$NORMAL	*****	X	04
SS\$NOSUCHNODE	*****	X	04
SS\$REJECT	*****	X	04
SS\$VCBROKEN	*****	X	04
TIMEOUT	000003BB	R	04
TQESB_RQTYPE	= 0000000B		
TQESB_TYPE	= 0000000A		
TQESC_SSSNGL	= 00000001		
TQESK_LENGTH	= 00000030		
TQESL_FPC	= 0000000C		
TQESL_FR3	= 00000010		
TQESL_FR4	= 00000014		
TQESW_SIZE	= 00000008		

+-----+
! Psect synopsis !
+-----+

PSECT name	Allocation	PSECT No.	Attributes
ABS	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
\$AB\$\$	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
\$\$\$040	00000010 (16.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
\$\$\$002	000000F0 (240.)	03 (3.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
\$\$\$100	000006B7 (1719.)	04 (4.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

+-----+
! Performance indicators !
+-----+

Phase	Page faults	CPU Time	Elapsed Time
Initialization	32	00:00:00.04	00:00:02.20
Command processing	110	00:00:00.40	00:00:02.23
Pass 1	412	00:00:10.56	00:00:34.95
Symbol table sort	0	00:00:01.38	00:00:03.59
Pass 2	346	00:00:03.37	00:00:09.25
Symbol table output	25	00:00:00.13	00:00:00.34
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	929	00:00:15.90	00:00:52.58

The working set limit was 1800 pages.

91243 bytes (179 pages) of virtual memory were used to buffer the intermediate code.

There were 80 pages of symbol table space allocated to hold 1267 non-local and 113 local symbols.

2014 source lines were read in Pass 1, producing 26 object records in Pass 2.

34 pages of virtual memory were used to define 31 macros.

+-----+
! Macro library statistics !
+-----+

Macro library name	Macros defined
\$255\$DUA28:[SYSLOA.OBJ]CLUSTER.MLB;1	3
\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	17
\$255\$DUA28:[SYSLIB]STARLET.MLB;2	6
TOTALS (all libraries)	26

1394 GETS were required to define 26 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:CNXMAN/OBJ=OBJ\$:CNXMAN MSRC\$:CNXMAN/UPDATE=(ENH\$:CNXMAN)+EXECML\$/LIB+LIB\$:CLUSTER/LIB

0392 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

